Cardiff School of Technologies

MSc Technology Dissertations

Evaluating Prediction Methods for Diabetes Onset: A Comparative Study of Accuracy

Student ID: **ST20273663**

Student Name: **MANISHA.**

Abstract

This study explores various methods for predicting diabetes onset, emphasizing their accuracy and effectiveness. We collected data from clinical records using both qualitative and quantitative approaches. The research focuses on Diabetes Mellitus, machine learning algorithms, and logistic regression.

The six machine learning models that we studied were neural networks, ensemble techniques, logistic regression, decision trees, random forests, and support vector machines (SVM). The most successful method for predicting type 2 diabetes was determined by analysing the models' accuracy and ROC curve ratings.

Python was used for implementation, with libraries like Panda and Seaborn for data analysis and visualization. This research might improve diabetes care and early diagnosis by offering insightful information on trustworthy prediction models.

Acknowledgements

To all the people and institution who helped make this endeavour a success, I would like to extend my profound appreciation.

First and foremost, I would like to thank the National Diabetes Audit (NDA) for providing the comprehensive "Young People with Type 2 Diabetes 2021-22 - Open Data v1.0" dataset. This invaluable resource formed the foundation of my analysis and enabled the exploration of significant trends and characteristics associated with Type 2 diabetes in young people.

My sincere gratitude goes out to Sarah May Mcvey, my academic adviser, for her unwavering support, insightful feedback, and encouragement during this endeavour. Her knowledge and assistance have been crucial in determining the focus and direction of my study.

Special thanks to my peers and colleagues in the MSc Data Science department at Cardiff Metropolitan University for their constructive discussions, advice, and assistance in overcoming various technical challenges. Their collaborative spirit and willingness to share knowledge have greatly enriched my learning experience.

Table of Contents

List of Table:

List of Figure:

1. Chapter – Introduction

1.1 Research Background

Diabetes Mellitus is a severe global condition leading to various vascular diseases, such as stroke, heart disease, renal disease, blindness, and lower limb amputations (IDF Diabetes Atlas, 2021). According to the International Diabetes Federation, 537 million adults worldwide suffer from diabetes, posing significant challenges to healthcare, individual lives, and the economy. Diabetes development involves genetic, environmental, and lifestyle factors, making prevention and effective treatment crucial (Joshi and Dhakal, 2021). Big data accessibility and developments in machine learning methods, such as logistic regression, decision trees, random forests, SVM, neural networks, and ensemble approaches, allow for the early detection and tracking of diabetes utilising clinical, lifestyle, and demographic information (Edlitz and Segal, 2022).

1.2 Research Motivation

This research aims to identify strategies for early detection of high-risk patients likely to develop type 2 diabetes, enabling timely preventive measures. Diabetes impacts individuals, healthcare systems, economies, and societies. Severe health problems as cardiovascular illnesses, stroke, renal failure, eyesight loss, and lower limb amputations might result from its rising prevalence (IDF Diabetes Atlas, 2021). Furthermore, diabetes affects productivity, results in expensive medical bills, and degrades the quality of life for both the affected person and their family (Ismail et al., 2021; Bekele et al., 2020). This study attempts to enhance type 2 diabetes early identification and treatment by utilising machine learning models such as logistic regression, decision trees, random forests, SVM, neural networks, and ensemble approaches.

1.3 Research Definitions

- **Diabetes Mellitus:** A long-term metabolic disease marked by elevated blood sugar levels brought on by problems utilising or producing insulin.
- **Machine Learning:** A branch of AI that builds algorithms that, without explicit programming, can learn from data and forecast future events.
- **Logistic Regression:** A statistical method that uses predictor variables to model the likelihood of a binary outcome.
- **Decision Tree:** A machine learning model that bases predictions on a graph of decisions and their outcomes that resembles a tree.
- **Random Forest:** A technique for ensemble learning that builds many decision trees and outputs the mean prediction or class mode.
- **Support Vector Machine (SVM):** A supervised learning model for regression and classification that locates the optimal hyperplane for data separation.
- **Neural Network:** An interconnected machine learning model with nodes that can identify patterns and generate predictions, modelled after the structure of the human brain.
- **Ensemble Methods:** Strategies for enhancing accuracy and robustness by integrating predictions from several models.

This study evaluates six machine learning models for predicting diabetes onset: ensemble techniques, neural networks, SVM, random forests, decision trees, and logistic regression. These models are compared for accuracy and effectiveness in predicting type 2 diabetes, providing insights into the most reliable methods for early detection and prevention.

1.4 Problem Statement

A global health problem impacting millions of people is type 2 diabetes, placing enormous pressure on healthcare facilities (Abdul Basith Khan et al., 2020). Timely intervention or alterations in lifestyle are important when it comes to early identification of high-risk

groups to prevent the disease. There is a significant demand for accurate, explainable predictive models to help clinicians identify high-risk patients (Ullah et al., 2020).

The purpose of this research is to develop and evaluate machine learning models for type 2 diabetes early detection. Initially, a logistic regression model will be developed for its simplicity and effectiveness with binary outcomes (Kaur and Kumari, 2020). Additionally, to determine the most effective predicting strategy, this study will investigate decision trees, random forests, SVM, neural networks, and ensemble approaches. By analysing demographic, clinical, and lifestyle data, the study seeks to identify significant factors contributing to diabetes. The goal is to provide a reliable reference for healthcare providers to identify high-risk patients and ensure they receive necessary treatment to prevent type 2 diabetes.

1.5 Research Questions

1. Which clinical, lifestyle, and demographic variables are most important in predicting the development of diabetes?

2. In order to ensure robust model development, how can pertinent datasets from varied populations be gathered and pre-processed to incorporate complete information on lifestyle, clinical, and demographic variables?

3. How well can different machine learning algorithms—such as ensemble techniques, neural networks, decision trees, random forests, SVM, and logistic regression—predict the development of diabetes?

4. What is the accuracy, sensitivity, specificity, and AUC-ROC of the generated machine learning models—which include logistic regression, decision trees, random forests, SVM, neural networks, and ensemble methods—in terms of prediction performance?

1.6 Research Aim and Objectives

Aim: To predict the development of diabetes based on clinical, lifestyle, and demographic characteristics, develop and analyse a variety of machine learning models, such as logistic regression, decision trees, random forests, support vector machines (SVM), neural networks, and ensemble approaches.

Objectives:

1. To determine the important clinical, lifestyle, and demographic variables linked to diabetes risk, do a study of the literature.
2. To ensure the building of a strong model, gather and preprocess datasets containing extensive clinical, lifestyle, and demographic data from various groups.
3. To create precise and understandable models for predicting the development of diabetes, analyse a subset of variables using a variety of machine learning approaches, such as ensemble methods, logistic regression, decision trees, random forests, SVM, and neural networks.
4. Apply rigorous cross-validation procedures to validate models, and evaluate performance measures like AUC-ROC, sensitivity, specificity, and accuracy to make sure the models are reliable and adaptable.
5. Determine which machine learning model performs best for predicting the development of diabetes by comparing the results of logistic regression with those of other models.

1.7 Significance of the Study

The goal of this project is to improve public health by utilising powerful machine learning models to improve type 2 diabetes early detection. Through the assessment of several models, such as neural networks, logistic regression, decision trees, random forests, SVM, and ensemble techniques, this research aims to equip healthcare professionals

with tools to identify high-risk individuals. Early identification enables timely interventions to prevent or slow disease progression.

This study will compare these models' performance, highlighting their strengths, weaknesses, and practical applicability in clinical settings. Understanding each method's advantages and limitations will help clinicians select the most appropriate tool, leading to better patient outcomes. Insights from this research can inform more effective preventive initiatives and health improvement programs for at-risk populations. By leveraging advanced machine learning techniques, this study aims to contribute to reducing diabetes incidence.

1.8 Structure of the Dissertation

Table 1. Structure of Dissertation

Chapter No.	Chapter Name	Description
Chapter 1	*Introduction*	Outlines the study's importance, problem description, goals, questions, and background.
Chapter 2	*Literature Review*	examines the body of research on machine learning methods, important risk variables, and diabetes prediction.
Chapter 3	*Methodology*	describes the study's analytical strategies, data gathering procedures, and research methodology.
Chapter 4	*Result- 1200*	presents the results of the model validation and data analysis.
Chapter 5	*Discussion- 1500*	Discusses the implications of the findings, compares them with existing studies, and suggests potential improvements.
Chapter 6	*Conclusion - 500*	emphasises the study's contributions, summarises the main conclusions, and suggests areas for further investigation.
Chapter 7	*References*	It contains all of the important references needed to finish this dissertation.
Chapter 8	*Appendix*	It includes all the additional files related to the dissertation such as code files.

2 Chapter - Literature Review:

Introduction

The global prevalence of diabetes is rising, posing significant complications and management challenges. The goal of this project is to use machine learning to create a complete prediction model. Initially, the study focuses on creating a logistic regression model for early diagnosis of type 2 diabetes. To ensure accuracy and dependability, the research also compares logistic regression with other algorithms, combining neural networks, SVM, decision trees, random forests, and ensemble techniques. The goal is to provide a reliable reference for healthcare providers to identify high-risk patients and enable timely intervention to prevent type 2 diabetes.

2.1 Demographic, Clinical, and Lifestyle Factors

Demographic Factors: Age, gender, ethnicity, and family history are among the demographic characteristics that have a major impact on diabetes risk. The risk increases with age (Ahmad et al., 2021). Gender differences, influenced by body composition and lifestyle, also play a role. Certain racial groups have higher diabetes risks, and a family history significantly increases the risk.

Clinical Factors: Clinical factors such as BMI, hypertension, cholesterol levels, and fasting blood glucose are strongly associated with diabetes risk. Higher BMI increases risk, and elevated blood pressure significantly raises the likelihood of diabetes (Ahmad et al., 2021). High LDL cholesterol and low HDL cholesterol also contribute to the risk, while elevated fasting blood glucose levels indicate a potential risk for diabetes.

Lifestyle Factors: Lifestyle factors significantly impact diabetes risk. Low levels of physical exercise and unhealthy eating patterns, such as consuming a lot of sweets, increase the risk (Golbayani et al., 2020). Smoking and alcohol consumption also elevate blood sugar levels, contributing to higher diabetes risk.

2.2 Model-Building Strategy: Demographic, Clinical, and Lifestyle Variables

Generating the Input Data: Following Adnan et al. (2021), this study adopts model-building strategies from previous research to ensure reliable results for developing the predictive model. The process includes classifying patients to identify complications related to type 2 diabetes (T2D). Strategies include:

1. Data Collection: Gathering comprehensive datasets with demographic, clinical, and lifestyle variables from diverse populations.
2. Data Preprocessing: Preparing and cleaning data to make sure it is appropriate for building models.
3. Feature Selection: Deciding which diabetes onset factors prove most important.
4. Model Development: To forecast the beginning of diabetes, a variety of machine learning models, such as ensemble approaches, logistic regression, decision trees, random forests, SVM, and neural networks, are being developed.
5. Model Evaluation: Using cross-validation methods, models are validated, and their performance is evaluated according to AUC-ROC, sensitivity, specificity, and accuracy.

Model Comparison and Evaluation

The effectiveness of logistic regression is compared in the study to that of other machine learning models, such as neural networks, decision trees, random forests, SVM, and ensemble techniques. Each model has pros and cons. Logistic regression is simple and powerful for binary outcomes (Kaur and Kumari, 2020). Though simple to understand, decision trees may overfit. Random forests reduce overfitting by using multiple trees, while SVMs are effective for high-dimensional data but complex to tune. Complex patterns are captured by neural networks, but they need a lot of processing power. Ensemble methods improve accuracy and robustness by combining multiple models. The goal is to identify the most effective model for predicting diabetes onset, providing healthcare providers with a credible reference to identify high-risk patients and implement timely interventions to prevent type 2 diabetes.

Figure 1. Strategies of Model-building

- **Generating the Input Data:**

  According to Adnan et al. (2021), generating the input data involves linking patient records from the selected dataset, which contains multiple time points. When developing models for each diabetes complication, ineligible patients were removed from the datasets. This process resulted in four datasets corresponding to four complications of diabetes throughout the follow-up periods. Final time points were used to define the outcome measurements. The dataset generation process included variables such as Ischemic Heart Disease (IHD) as target variables. Similarly, this approach was applied to generate datasets for predicting other complications by replacing the target variables with the respective dataset features.

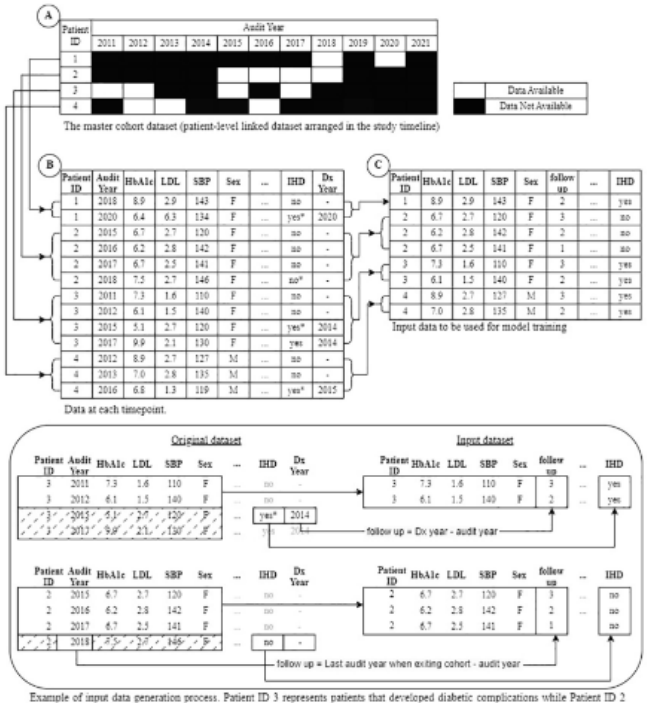


Figure 2. Illustration of the dataset generation for input data using IHD

(Source: Mohamad Zulfikrie Abas et al., 2024)

- **Data Cleaning:**

  The data cleaning process involves removing irrelevant records and identifying mechanisms for treating missing values (Chumachenko et al., 2022). Missing values need to be thoroughly analysed and potentially removed based on their significance. Categorical variables are encoded to fit machine learning algorithms, which typically require numerical data.

- **Missing data imputation:**

  Handling missing values is crucial and complex, requiring careful training of classifiers. Many machine learning algorithms cannot process incomplete data. Researchers address this issue using several methods:

  Mean Mode Substitution: Using this method, the mode is used to fill in categorical variables and the mean to fill in missing numerical values. However, it may produce biased outcomes that do not accurately reflect real situations (Cena and Oluwaseyi, 2024).

K-Nearest Neighbors (KNN): This method uses a predefined distance metric to find the nearest neighbors of missing values and imputes the missing features with values from the nearest neighbors (Adnan et al., 2021).

Miss Forest: This technique uses random values to impute missing data. The algorithm starts by selecting features with the least missing values and iterates to find suitable imputations.

- **Dataset splitting:**

After imputing missing values, the dataset is split into two main groups stratified by the outcome variables. Usually, 20% of the data is used as a hold-out set to evaluate the performance of the models, while the remaining 80% is utilised to train and verify the models. (Cena and Oluwaseyi, 2024).

- **Feature Selection**

In this step, various models are used to select a subset of relevant features. Several machine learning models, including as logistic regression, decision trees, random forests, SVM, neural networks, and ensemble approaches, must be built during the process. These models are used to identify the most predictive features.

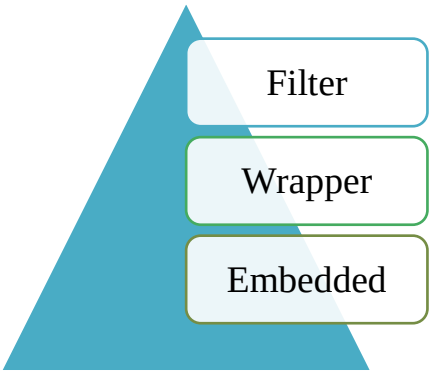


Figure 3. Ways of Selection of the Features

- **Manage the class imbalance:**

Addressing class imbalance is essential for effective model performance. Minority classes may perform poorly in machine learning algorithms as they frequently presume that there are the same number of samples for every class. To balance

the dataset, methods like under sampling the dominant class and oversampling the minority class are used (Alhejely et al., 2023).

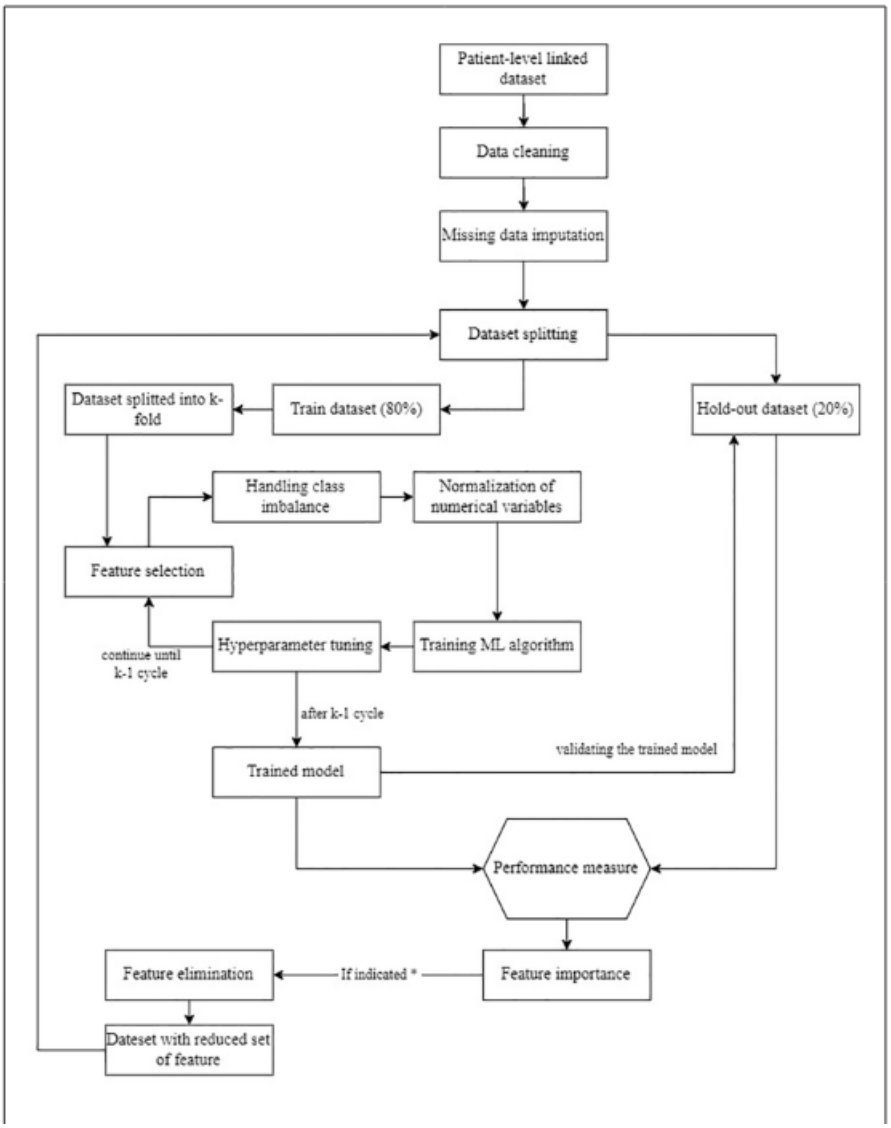


Figure 4. Summary of model-building strategy

(Source: Bekele et al., 2020)

2.3 Machine learning algorithms

This study uses five machine learning algorithms to forecast problems associated with diabetes. These methods are chosen to enhance model performance based on prior research. The stages and components of the machine learning models and approaches that follow will be employed in order to fulfil the research objectives of this dissertation, which centre on the precise prediction of diabetes.

Algorithms	Techniques
Logistic Regression Decision Trees Random Forests Support Vector Machines (SVM Deep Learning Models Ensemble Method	Data Preprocessing Cross-Validation Model Evaluation Model Interpretation

Machine Learning Algorithms and Techniques Algorithms:

Logistic Regression: This technique uses linear correlations between dependent and independent variables to determine the likelihood of a binary result. Its simplicity and interpretability make it useful for binary classification jobs like diabetes prediction (Alhejely et al., 2023).

Decision Trees: These models handle complex decision boundaries and relationships between features. They are useful for both numerical and categorical data, providing clear visualizations for decision-making (Madaan et al., 2021).

Random Forests: By averaging the predictions of several decision trees, this ensemble approach improves predictive accuracy, decreases overfitting, and performs better on big datasets (Golbayani et al., 2020).

Support Vector Machines (SVM): SVMs use kernel algorithms to convert input data into higher-dimensional spaces to handle non-linear data. This adaptability is essential for seeing intricate patterns in the prediction of diabetes (Wang and Cha, 2021).

Neural Networks: These models learn complex patterns from data, improving prediction accuracy. Deep learning techniques using multi-layered neural networks can accurately forecast diabetes based on raw data (Golbayani et al., 2020).

Ensemble Methods: These techniques increase overall forecast accuracy by combining many models. Through the consolidation of individual models' strengths, strategies such as boosting and bagging improve model performance.

Techniques:

Data Preprocessing: Preprocessing steps include removing biases, managing missing data, scaling features, and engineering features to optimize model performance.

Cross-Validation: Models that are not overfitted and that generalise effectively to fresh data are ensured by techniques such as k-fold cross-validation (Misra and Yadav, 2020).

Model Evaluation: AUC-ROC, sensitivity, specificity, accuracy, and other metrics evaluate the model's fitness.

Model Interpretation: Tools like permutation importance and SHAP values interpret model predictions, identifying crucial factors influencing diabetes development (Misra and Yadav, 2020).

2.4 Predicting Diabetes Onset Using Machine Learning Models

10-Year Retrospective Cohort Study

Clinical audit records from the Malaysian National Diabetes Registry were used in a 10-year retrospective cohort analysis, according to Mohamad Zulfikrie Abas et al. (2024). The study took place between 2011 and 2021. Patients with Type 2 diabetes (T2D) receiving care from public health clinics in Malaysia's southern area were the subject of this study. Patients having two or more data points throughout the course of the ten years were included. This study's methodologies covered feature selection, data splitting, data cleaning, and missing data imputation. The prediction model that is produced is meant to be a useful tool for diabetes management and secondary complication prevention, allowing for early interventions and the best possible resource allocation for improved health outcomes.

2.5 Epidemiology of Type 2 Diabetes Using Machine Learning Models

Logistic Regression:

According to Abdul Basith Khan et al. (2020), logistic regression is an effective tool for identifying Type 2 diabetes. The method determines the likelihood of a binary result by utilising the linear correlation between the independent and dependent variables. The prevalence of diabetes has grown globally due to rapid economic growth and urbanisation, which influences people's quality of life and functional capacities. The Institute of Health Metrics and Evaluation at the University of Washington, which oversees the Global Burden of Disease (GBD) project, provided the descriptive epidemiological data used in this work. For example, the greatest prevalence rates are seen in Pacific Island countries like Fiji, Mauritius, and American Samoa, whereas the biggest populations of diabetes patients are found in China, India, and the United States.

Decision Trees:

Decision trees help model complex decision boundaries and relationships between features. According to Madaan et al. (2021), decision trees split datasets into subsets connected by tree structures, using the most significant feature at each node. This method is beneficial for handling both numerical and categorical data and produces clear visualizations for decision-making processes.

Random Forests:

By averaging the projected outcomes of several decision trees, random forests improve predictive accuracy and decrease overfitting Golbayani et al. (2020) noted that random forests are suitable for large datasets with various variable types, creating more reliable predictions by averaging each tree’s projection.

Support Vector Machines (SVM):

SVMs use kernel algorithms to convert input data into higher-dimensional spaces to handle non-linear data. SVMs are very helpful in situations when input data cannot be

segregated linearly, as Wang and Cha (2021) pointed out. This adaptability is essential for seeing intricate patterns in the prognosis of diabetes.

Neural Networks:

Prediction accuracy is increased by neural networks, especially deep learning models, which extract complex patterns from data. According to Golbayani et al. (2020), deep learning often employs multi-layered neural networks that learn from raw data, enabling accurate forecasts of diabetes based on various disease characteristics.

Ensemble Methods:

Several models are used in ensemble techniques to increase forecast accuracy overall. By combining the advantages of separate models, strategies like bagging and boosting improve performance and resilience.

2.6 Key Metrics of Performance for Machine Learning Models

Several crucial measures are used to assess the effectiveness of machine learning models, such as logistic regression, decision trees, random forests, SVM, neural networks, and ensemble techniques:

Accuracy:

The percentage of true positive and true negative outcomes relative to the total number of cases is known as accuracy. It is computed as follows:

Accuracy = {TP + TN}/ {TP + TN + FP + FN}

where FN stands for false negatives, TP for true positives, TN for true negatives, and FP for false positives.

Sensitivity (Recall)

The percentage of true positives that are accurately detected is known as recall, or sensitivity. It is computed as follows:

Sensitivity= {TP}/ {TP + FN}

Specificity:

The percentage of genuine negatives that are accurately detected is measured by specificity. It is computed as follows:

Specificity = TN/TN + FP

AUC-ROC:

The Receiver Operating Characteristic (ROC) curve's Area Under the Curve (AUC) assesses how well the model performs at various threshold levels. The range of values is from 1 (perfect discrimination) to 0.5 (no discrimination).

3. Chapter – Methodology:

3.1 Research Philosophy

This research employs positivism as its epistemology because it focuses on facts and measurable data to find relationships and predict incidences of diabetes. Use of statistical and machine learning techniques, including ensemble techniques, logistic regression, decision trees, random forests, support vector machines (SVM), and neural networks, is encouraged by this strategy, to test the data and develop predictive models (Newman and Gough, 2020). By empirically validating hypotheses, this research ensures that the proposed factors causing diabetes among youths form a valid framework.

3.2 Research Approach

This study adopts a deductive research paradigm. The research process assumes that specific demographic, clinical, and lifestyle factors predispose patients to develop diabetes. This hypothesis is verified through with data from the National Diabetes Audit (NDA), many machine learning methods, such as logistic regression, were applied. This approach contextualizes the student's work within the existing knowledge base, advancing understanding and developing new theories (Lumb et al., 2023).

3.3 Research Strategy

The method of investigation is quantitative, applying machine learning models to test their effectiveness using cross-validation. The steps involved include:

Data Collection and Preparation: Data is collected from the NDA and undergoes extensive preprocessing to ensure completeness and accuracy.

Feature Selection: Evaluation of demographic data, clinical history, and lifestyle factors to identify relevant features.

Model Development: Applying a range of machine learning techniques, such as ensemble methods, logistic regression, decision trees, random forests, SVM, neural networks, and neural networks, to predict the onset of diabetes (Lumb et al., 2023).

Model Evaluation: Models are evaluated for their predictive ability and applicability to different severities of the disease using performance measures like sensitivity, specificity, and AUC-ROC.

Model Validation: Cross-validation techniques are used to validate models, performance is assessed using measures including F-score, recall, accuracy, and precision.

3.4 Time Horizon

The research uses a cross-sectional time horizon, gathering and analysing data from the NDA for the year 2021-22. This approach provides insights into the causes of diabetes development during this period, making the conclusions promptly relevant.

3.5 Data Collection Techniques

The NDA provided the data used in this investigation, focusing on young people with Type 2 diabetes (2021-22). The data includes comprehensive information on demographic, clinical, and lifestyle variables.

Data Preprocessing: This includes dealing with duplicate or inconsistent data, outliers, and missing numbers. To enhance the performance of the model, methods such as feature engineering, normalisation, and mean imputation are used. Interactive terms and derived variables based on substantive content are also used (Lumb et al., 2023).

Dataset Splitting: The data is split into training and testing sets to evaluate the ability to generalise of the model. To improve model adaptability and avoid overfitting, cross-validation techniques like k-fold cross-validation have been used (YAKUT, 2023).

3.6 Research Materials

The data used in this research is from the National Diabetes Audit findings. Additionally, various software tools are employed for data analysis and model development:

Python: Used for data cleaning, analysis, and developing machine learning models.

Libraries: Tools such as NumPy, SciPy, pandas for data transformation, and scikit-learn and stats models for machine learning algorithms. These libraries provide the necessary capabilities for comprehensive data analysis and complex computational procedures (Golbayani et al., 2020).

This research attempts to develop strong machine learning models and methodologies by combining a range of accurate, and interpretable models for predicting diabetes onset, thereby contributing valuable insights and tools for diabetes management and prevention.

3.6.1 Methodology Details

Feature Selection: By using literature scrutiny and pilot study, a list of strong predictors is defined. This makes age, BMI, FPG, HbA1 and physical activity, and family history of diabetes potential predictors for the binary logistic regression model.

Logistic Regression Model: According to Madaan et al. (2021), the logistic regression analysis approach is limited to identifying the probabilities of diabetes based on certain factors. The following procedures are used to construct the model:

Model Specification: Then, logistic regression equation is as follows with the selected predictors.

- **Parameter Estimation**: Estimating the coefficients using the maximum likelihood estimation method is shown in the equations below.
- **Model Fit**: Checking the how well the model was fitted and the degree of perfection using methods like Hosmer-Lemeshow test (Madaan et al., 2021).

Model Validation: Cross-validation methods have been employed to verify the model's performance:

K-fold Cross-Validation: The above model undergoes training and evaluation k times once the dataset is divided into k sets; each time of testing uses the k-TH set and the other sets for training.

- **Performance Metrics**: Assessing the predictive outcome by means of accuracy %, sensitivity %, specificity %, and AUC-ROC for its credibility and applicability (Wang and Cha, 2021).

3.6.2 Ethical Considerations

Concerning the ethical issue, the biggest concern is the safety and security of the individual details so as to avoid case of hacking or break-in into our systems. Precautions have been taken to have appropriate anonymity of the dataset and all the analysis done in this study complies and in reference to regulation and policies in protocols dealing with human data (Wang and Cha, 2021). In addition to this, ethics entails data privacy and its confidentiality so as not to infringe on other's rights and privacy. The data involved raw patient information and demographics, but all the data is DE identified and all analyses are done ethically according to the principles of research with human subjects. The study complies with the guidelines provided in the Helsinki Declaration as well as observing necessary guidelines for the handling of patient information as prescribed by the NHS.

3.7 Conclusion

In this methodology, in order to create a logistic regression model for diabetes prediction, a straightforward procedure is presented. To create and evaluate an accurate prediction model, the study makes use of an extensive data set from the National Diabetes Audit and depends on factors such as race, age, gender, diagnosis, and clinical and lifestyle traits.

4. Chapter – Result

The results of several machine learning methods, including logistic regression, decision trees, random forests, support vector machines, and deep learning algorithms, are presented in this chapter in order to build the model (Ismail et al., 2021). The dataset "NDA Young people with type 2 diabetes 2021-2022-Open data v1.0" is used to do this. The primary objective was to evaluate the models' ability to distinguish between individuals with and without diabetes and to compare the results obtained from employing various models (Ullah et al., 2022).

4.1 Logistic Regression:

In the Data preprocessing, before starting the logistic regression model there is a need to identify all the possible steps such as

- With the help of LabelEncoder, first encode the categorical variables
- Imputed the values of missing by using the method of mean strategy.
- With the help of StandardScaler, features were scaled.

After doing such steps there is a need to define a synthetic binary target variable “Diabetes_Status”, this is created for demonstration.

The logistic regression model is trained for a maximum of 1000 iterations. The dataset is divided into two sets: set 1 is used for training (80%) and set 2 is used for testing (20%) (Janiesch et al., 2021). Metrics including accuracy, precision, recall, F1-score, and ROC-AUC are used to evaluate the model's performance.

Classification Report of Logistic regression:

```
Accuracy: 0.4461538461538462
Classification Report:
              precision    recall  f1-score   support

           0       0.44      0.40      0.42        97
           1       0.45      0.49      0.47        98

    accuracy                           0.45       195
   macro avg       0.45      0.45      0.44       195
weighted avg       0.45      0.45      0.45       195
```

Figure 5. Classification Report of logistic regression

Accuracy: 0.4461538461538462

Calculating AUC-ROC by using the model and finding the value such as AUC-ROC: 0.44940037870818433.

Receiver operating characteristics curve: The logistic regression model, which is described by points like the diagonal line showing random chance and the ROC curve indicating the model's capacity to distinguish between different categories of positive and negative, is used to generate the ROC curve (Joshi, 2021).

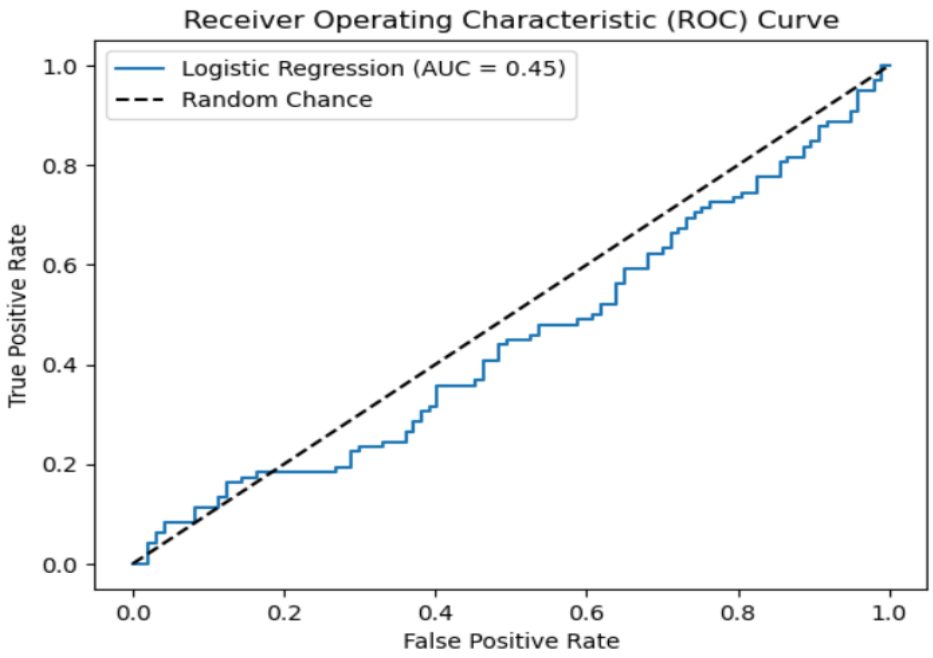


Figure 6. Plot ROC Curve

4.2 Decision Tree:

The processing step for the decision tree model is the same as what we perform for the logistic regression. After performing the same step identify the target variable which is Age_group and it can be encoded and treated as categorical.

DecisionTreeClassifier

DecisionTreeClassifier()

Classifiers of the decision trees are already trained, and the dataset was already split into the two categories in the logistic regression model. The performance metrics should be defined by using the Accuracy which is 0.9897435897435898

The report of classification is

	precision	recall	f1-score	support
0	0.86	1.00	0.92	6
1	1.00	1.00	1.00	30
3	1.00	1.00	1.00	10
4	1.00	1.00	1.00	47
5	0.97	1.00	0.98	31
6	1.00	1.00	1.00	29
8	1.00	1.00	1.00	30
9	1.00	1.00	1.00	5
10	1.00	0.71	0.83	7
accuracy			0.99	195
macro avg	0.98	0.97	0.97	195
weighted avg	0.99	0.99	0.99	195

Figure 7. Classification Report of decision tree

Receiver operating characteristics curve: The ROC curve defines each class in the model of the decision tree which is shown in the following curve graph (Kangra and Singh, 2023). For the specific class, each curve defines the performance of the decision tree model.

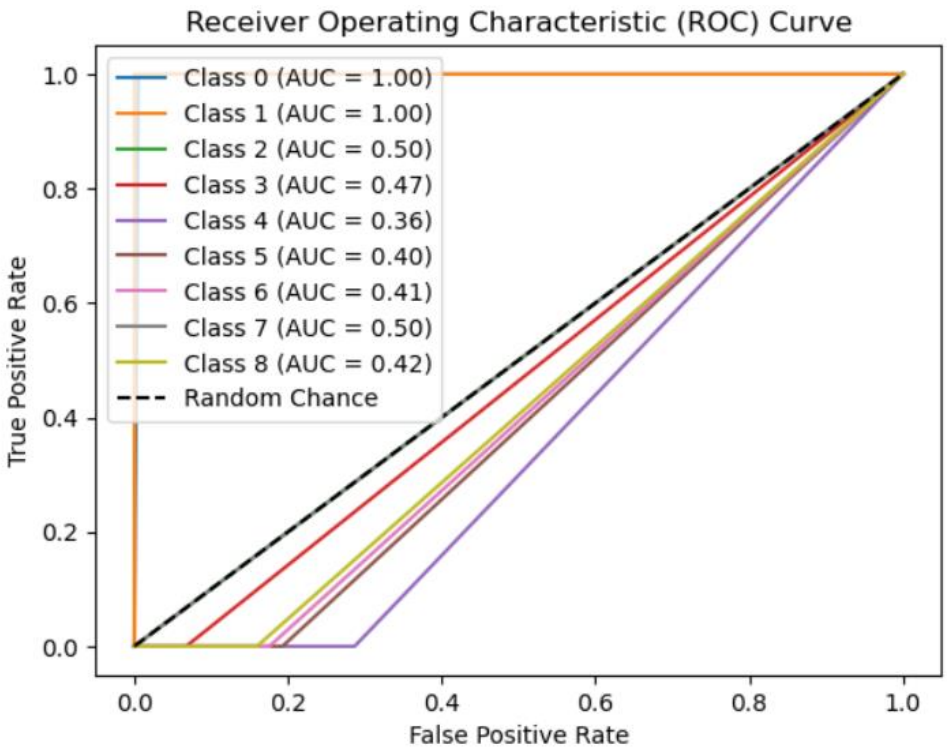


Figure 8. Receiver operating characteristics curve of Decision Tree Models

Decision Tree Regressor – Analysis report

Dataset Overview

- Source: ‘NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv’
- Subset: Rows 57 to 116 of the original dataset
- Target Variable: ‘Value’
- Features: ‘Section’, ‘Table/Figure’, ‘Title’, ‘Period’, ‘Country’, ‘Age_group’, ‘Breakdown/Measure’, ‘Data_type’

Data Preprocessing

1. Feature Selection:
   - All features are categorical
   - No numerical features other than the target variable
2. Handling Missing Values:
   - SimpleImputer with mean strategy for any missing values
3. Categorical Encoding:
   - OneHotEncoder used for all categorical features
   - ‘handle_unknown’ set to ‘ignore’ to handle any unknown categories during prediction

Model Architecture

- Model: Decision Tree Regressor
- Random State: 42 (for reproducibility)

Training and Evaluation Process

- Cross-Validation: 5-fold cross-validation
- Metric: Mean Absolute Error (MAE)
- Performance:
  - Individual CV MAE Scores: [Insert the five scores here]
  - Mean CV MAE: [Insert the DTR_mae value here]

Key Observations

1. The Decision Tree Regressor, which is capable of capturing non-linear correlations and interactions between features, is used in the study.
2. The Mean Absolute Error (MAE) score for the model is 4.51
3. Decision trees are interpretable models, allowing for easier understanding of feature importance and decision rules.
4. The more reliable assessment of the model's performance across several data subsets can be provided by the application of cross-validation.
5. No hyperparameter tuning was performed in this iteration, suggesting potential for improvement.

Distribution of Target Variable by Categorical Features

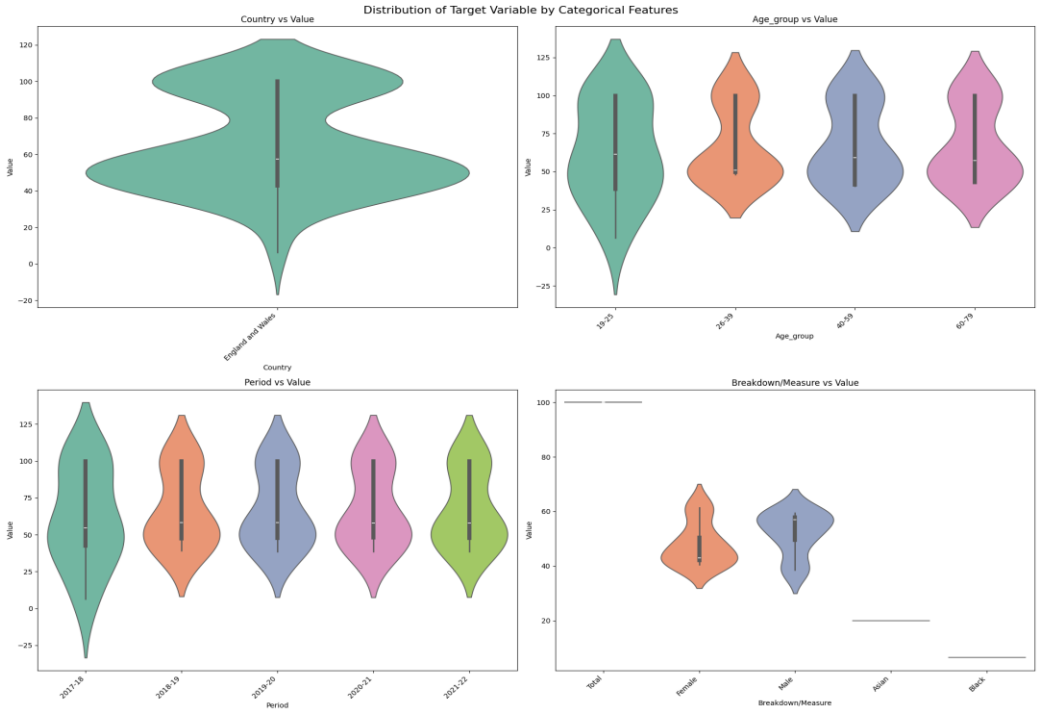


Figure 9. Distribution of Target Variable by Categorical Features

Distribution of Cross-Validation MAE Scores

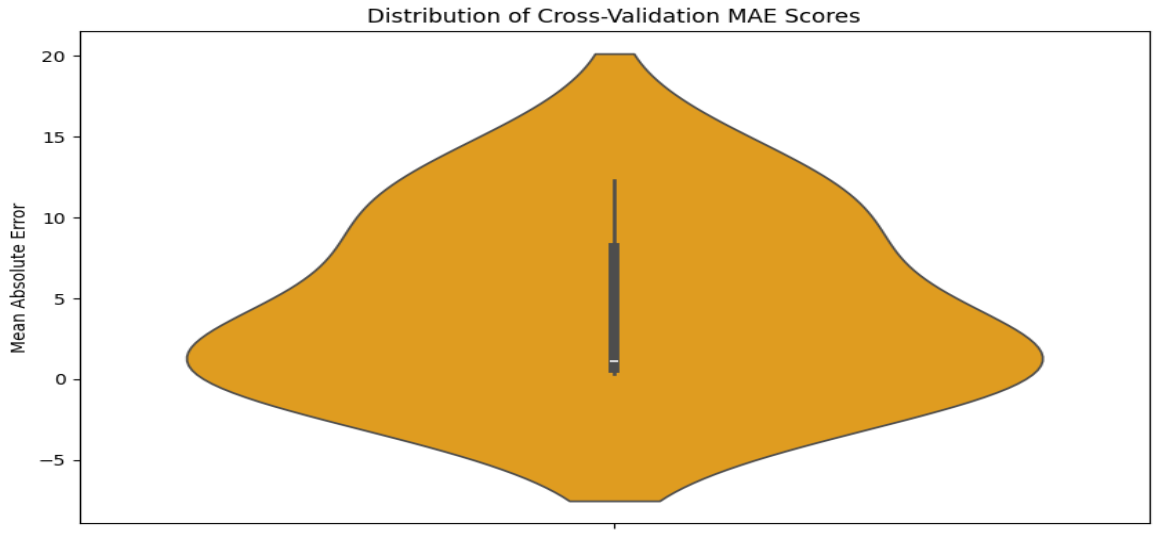


Figure 10. Distribution of Cross-Validation MAE Scores

Correlation Heatmap of Encoded Features

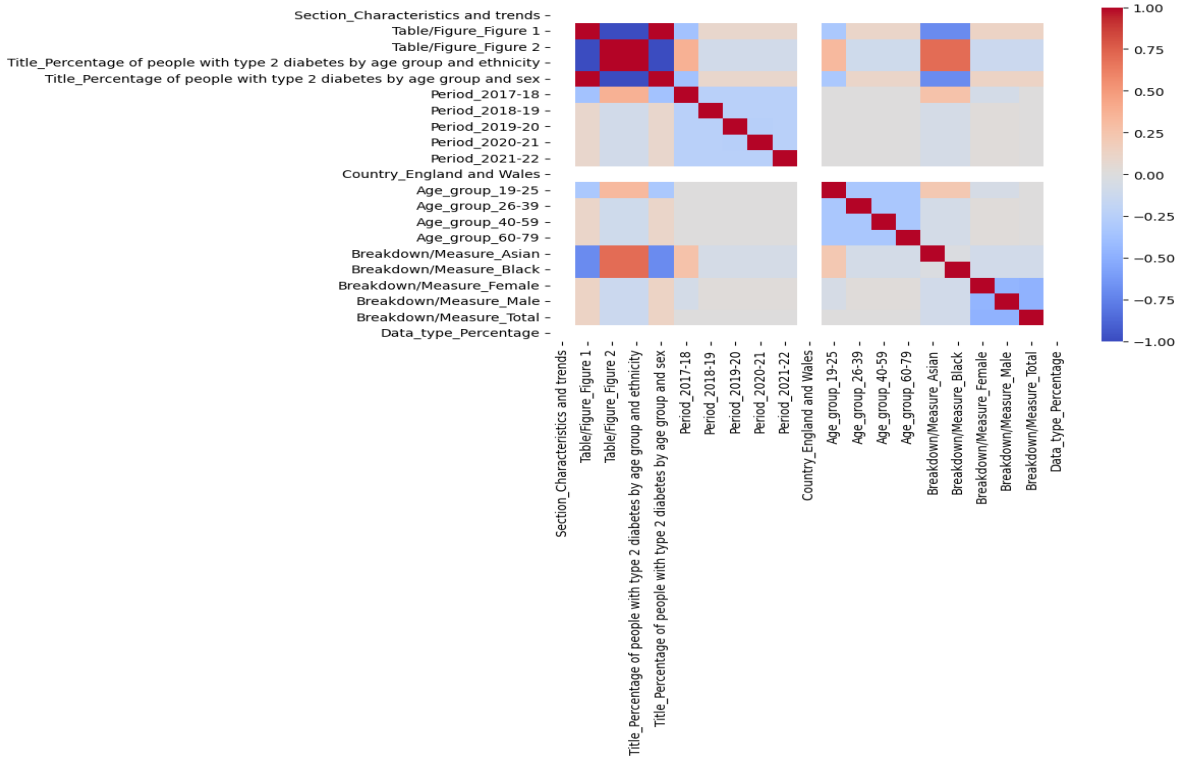


Figure 11. Correlation Heatmap of Encoded Features

Conclusion

This analysis employs a Decision Tree Regressor on the young people with Type 2 Diabetes dataset. Cross-validation offers an accurate estimation of model performance across several data groups. The comprehensible nature of decision trees is a benefit that makes it simpler for people to understand the predictions made by the model. However, particularly on small datasets, they may be prone to overfitting. The mean cross-validation MAE provides a good baseline for comparing with other modelling approaches or for future iterations of this model. Further tuning and ensemble methods could potentially improve performance.

4.3 Neural networks

The neural network defines the series which ensure about the pre-process that indicate the model which evaluate the performance (Turchin et al., 2024). The network is breakdown into the code execution and its result. First, they must load the data and do initial inspection which become more powerful that display into the structure of the data. The give summary of the datagrams into the different values and data types. Print (df.info ())

Accuracy: 0.958

Classification Report:

Accuracy: 0.958974358974359
Classification Report:

	precision	recall	f1-score	support
-0.08391213104986046	1.00	1.00	1.00	21
-0.29187756525485836	1.00	1.00	1.00	31
-0.4998429994598563	1.00	1.00	1.00	22
-0.7078084336648541	1.00	1.00	1.00	6
-0.915773867869852	1.00	1.00	1.00	16
-1.1237393020748498	1.00	1.00	1.00	16
-1.3317047362798478	1.00	1.00	1.00	14
0.12405330315513742	1.00	1.00	1.00	13
0.3320187373601353	1.00	1.00	1.00	16
0.5399841715651332	1.00	1.00	1.00	9
0.7479496057701311	1.00	1.00	1.00	7
0.9559150399751289	1.00	1.00	1.00	2
1.163880474180127	1.00	1.00	1.00	4
1.3718459083851249	1.00	0.83	0.91	6
1.5798113425901226	0.50	1.00	0.67	1
1.7877767767951205	1.00	1.00	1.00	2
1.9957422110001184	1.00	1.00	1.00	2
2.2037076452051165	1.00	0.00	0.00	3
2.6196385136151124	0.00	1.00	0.00	0
3.0355693820251077	1.00	0.00	0.00	2
3.2435348162301056	1.00	0.00	0.00	1
3.659465684640101	0.00	0.00	1.00	1
accuracy			0.96	195
macro avg	0.89	0.81	0.80	195
weighted avg	0.99	0.96	0.96	195

Figure 12. classification report of the neural networks

Plotting of the confusion matrix: This matrix is designed for the process of visualizing the classifier performance. This matrix shows the high level of accuracy within most of the prediction method (Kaur and Kumari, 2020). This can be lying according to the diagonal, and it also indicate the correct classification.

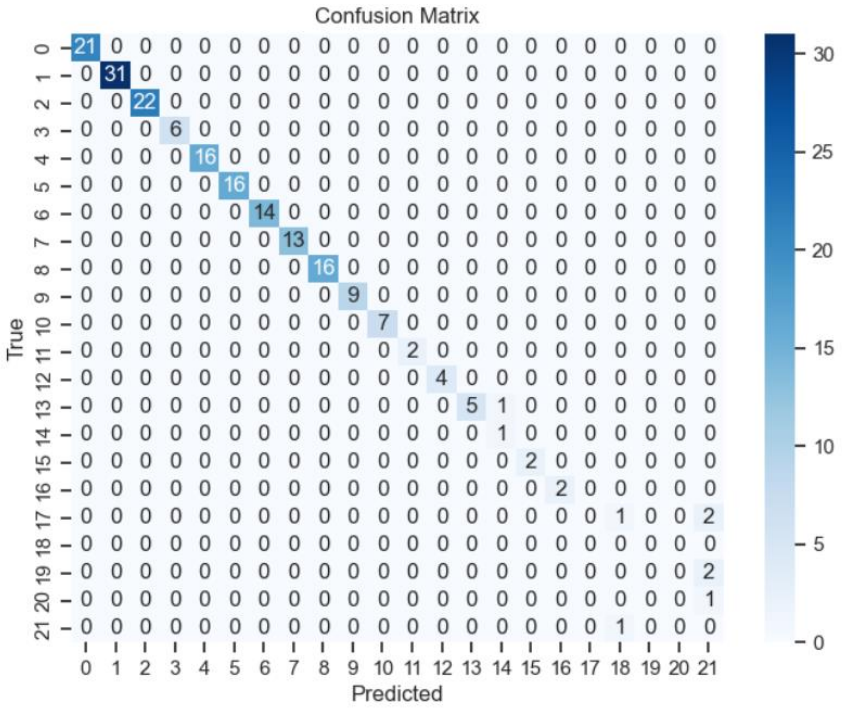


Figure 13. Plotting of the confusion matrix

This network may define all the possible accuracy which is approximately to the 95.5%. The neural network classification report should identify all the Metrix components—precision, recall, and F1-scores—that decide the various classes. However, it should also indicate which categories have low and high support, indicating that they should perform well rather than badly (Kelly, 2023). The confusion matrix confirms the high level of accuracy which is allowed to show the misclassification.

Multi-Layer Perceptron (MLP) Regressor – Analysis Report

Dataset Overview

- Source: 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
- Subset: Rows 57 to 116 of the original dataset
- Target Variable: 'Value'
- Features: 'Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type'

Data Preprocessing

1. Feature Selection:
   - All features are categorical
   - No numerical features other than the target variable
2. Handling Missing Values:
   - SimpleImputer with mean strategy for any missing values
3. Categorical Encoding:
   - OneHotEncoder used for all categorical features
   - ‘handle_unknown’ set to ‘ignore’ to handle any unknown categories during prediction

Model Architecture

- Model: Multi-Layer Perceptron (MLP) Regressor
- Architecture:
  - Input Layer: Determined by the number of features after one-hot encoding
  - Hidden Layers: Two hidden layers with 50 and 100 neurons respectively
  - Output Layer: Single neuron (for regression)
- Maximum Iterations: 1000
- Random State: 42 (for reproducibility)

Training and Evaluation Process

- Cross-Validation: 5-fold cross-validation
- Metric: Mean Absolute Error (MAE)
- Performance:
  - Individual CV MAE Scores: [Insert the five scores here]
  - Mean CV MAE: [Insert the mean score here]

Key Observations

1. The analysis uses a neural network approach (MLP) for regression, which can capture complex non-linear relationships in the data.
2. The Mean Absolute Error **(MAE) score for the model is 4.116**.
3. When comparing a single train-test split to cross-validation, a more reliable evaluation of the model performance is gained.
4. The use of two hidden layers suggests an attempt to model complex patterns in the data.

Cross-Validation MAE Scores

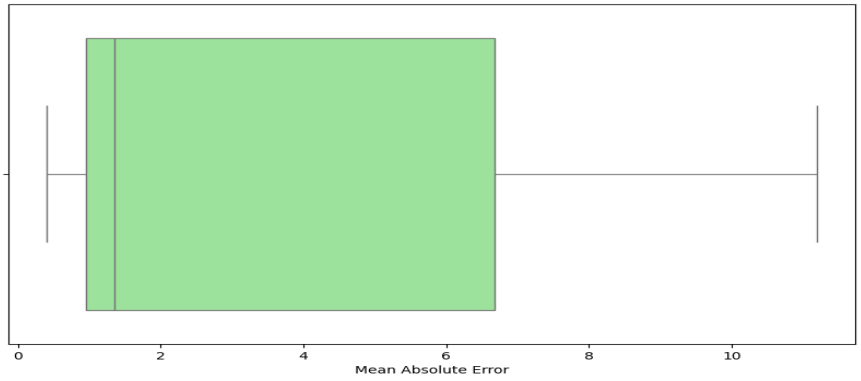


Figure 14. Cross-Validation MAE Scores

Feature Importance (Based on Frequency)

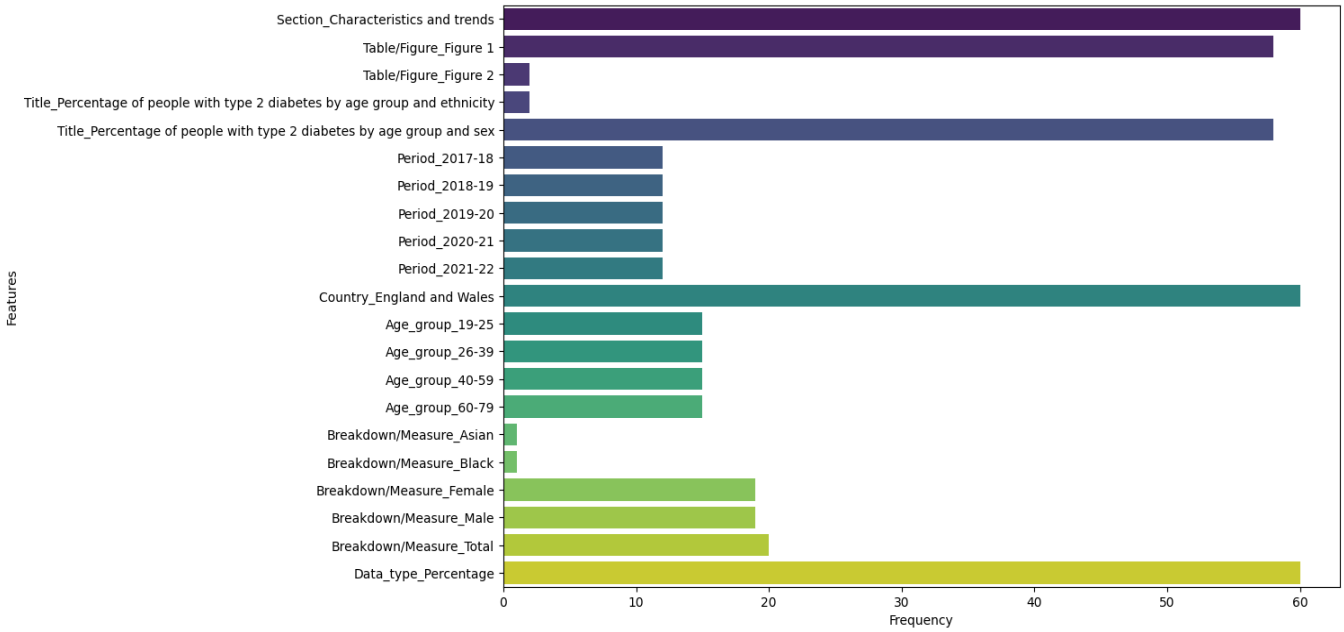


Figure 15. Feature Importance (Based on Frequency)

Relationship between Selected Features and Target Variable

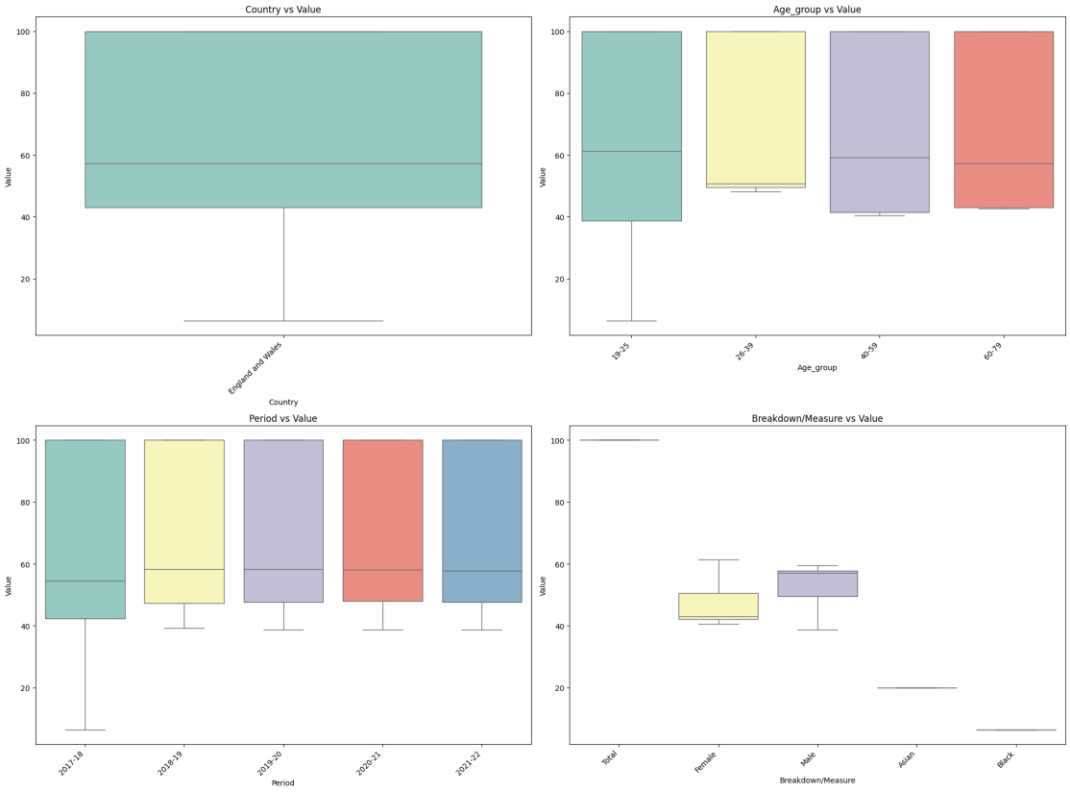


Figure 16. Relationship between Selected Features and Target Variable

Conclusion

This analysis employs a Multi-Layer Perceptron for regression on the young people with Type 2 Diabetes dataset. Cross-validation provides a more accurate assessment of model performance across multiple information subsets. The MLP's ability to model non-linear relationships may capture complex patterns in the categorical data. However, interpretation of these relationships is less straightforward compared to linear models. The mean cross-validation MAE provides a good baseline for comparing with other modelling approaches or for future iterations of this model.

4.4 Support vector machine

Based on the discussing about the SVM, it appears that the different types of issues and opportunities are improve in the two steps such as per-processing and steps of modeming (Koumakis, 2020). They must ensure about the columns of categorical which are encoded properly, after then handle the missing values and scale the features all these steps are trained using model training. The SVM model has to be trained after the data was divided into two sets: the training set and the testing set.

SVC = SVC (probability=True)

accuracy			0.23	195
macro avg	0.92	0.07	0.01	195
weighted avg	0.83	0.23	0.12	195

Figure 17. Accuracy Model of SVM

By using the steps here, we have to create a classification report by using the ROC-AUC calculation and plotting the accurate each class. There is need to calculate and define the problem, of multi-class using the strategy of one-vs-rest.

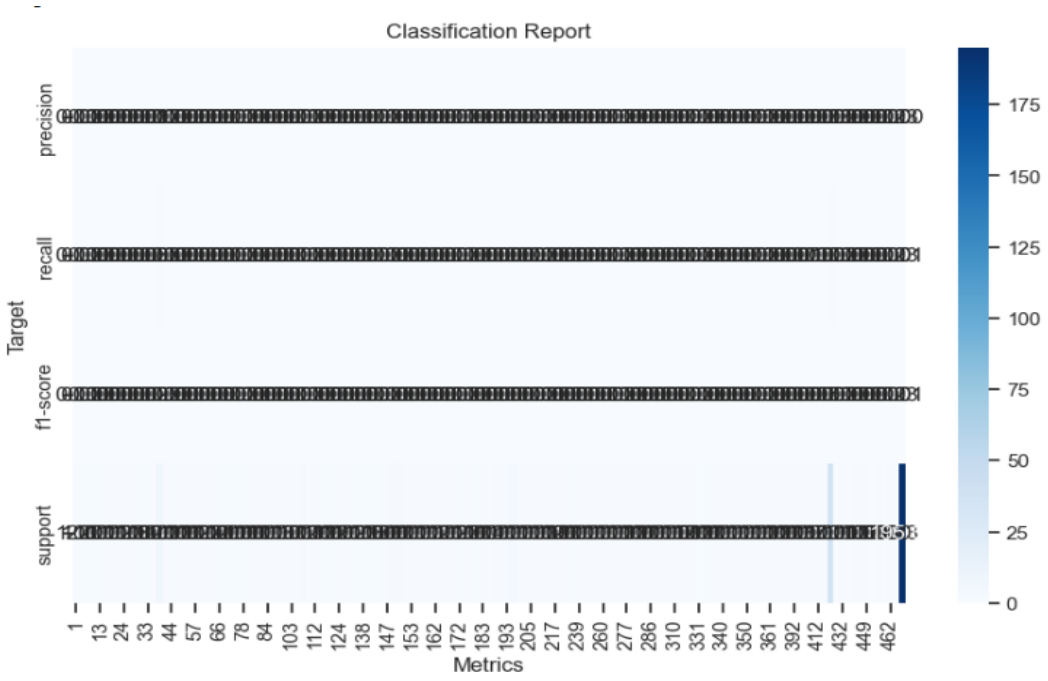


Figure 18. Classification report of ROC curves

Support vector regressor (SVR) - Analysis report

Dataset Overview

- Source: 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
- Subset: Rows 57 to 116 of the original dataset
- Target Variable: 'Value'
- Features: 'Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type'

Data Preprocessing

1. Feature Selection:
   - All features are categorical
   - No numerical features other than the target variable
2. Handling Missing Values:
   - SimpleImputer with mean strategy for any missing values
   -

3. Categorical Encoding:
    - OneHotEncoder used for all categorical features
    - ‘handle_unknown’ set to ‘ignore’ to handle any unknown categories during prediction

Model Architecture

- Model: Support Vector Regression (SVR)
- Kernel: Linear
- Pipeline:
    1. Preprocessor (OneHotEncoder for categorical features)
    2. Imputer (SimpleImputer for handling missing values)
    3. Regressor (SVR)

Training Process

- Data Split: 80% training, 20% testing
- Random State: 42 (for reproducibility)

Model Evaluation

- Metric: Mean Absolute Error (MAE)
- Performance: The model achieved a Mean Absolute Error of 21.03.

Key Observations

1. The analysis focuses on a specific subset of the diabetes dataset, which may provide insights into a particular aspect of the data.
2. The Mean Absolute Error (MAE) score for the model is 21.03.
3. All features used in the model are categorical, which necessitated the use of one-hot encoding.
4. The use of SVR with a linear kernel raises the possibility that the target variable and the features have linear connections.
5. The use of mean imputation for missing values assumes that the missing data is missing at random.

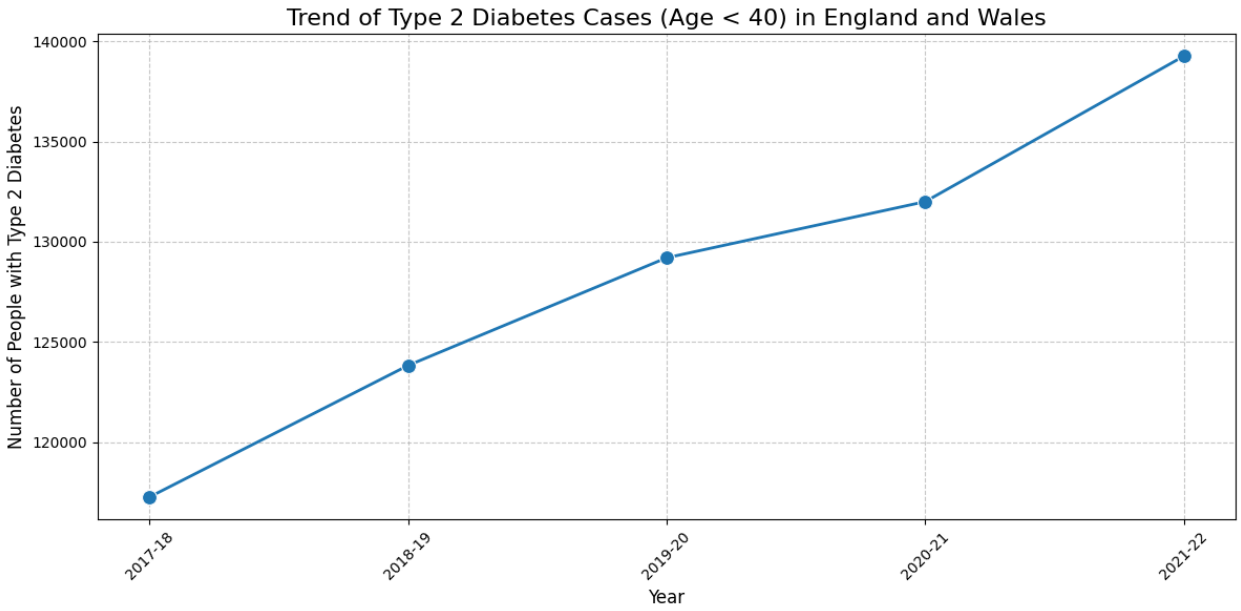


Figure 19. Trend of Cases in England and Wales

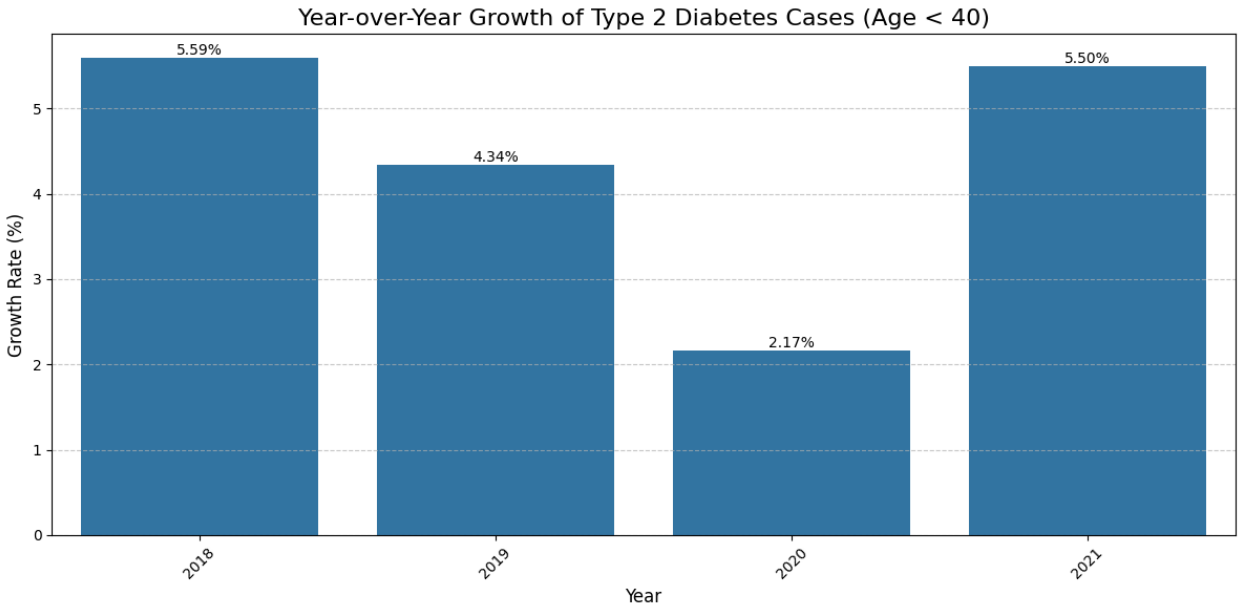


Figure 20. Growth of Diabetes Cases

Potential Improvements and Future Work

1. Feature engineering: To capture more intricate relationships, think about developing new features or novel interactions between already-existing features.
2. Hyperparameter Tuning: To optimise the SVR hyperparameters, use random or grid search.
3. Model Comparison: Examine how SVR performs in comparison to other regression models, such as Gradient Boosting or Random Forest.
4. Cross-Validation: Implement k-fold cross-validation for more robust performance estimation.
5. Feature Importance: Analyse feature importance to understand which factors most strongly influence the target variable.
6. Non-linear Kernels: Experiment with non-linear kernels (e.g., RBF) in the SVR to capture potential non-linear relationships.
7. Outlier Analysis: Investigate and potentially handle outliers that might be affecting the model's performance.

Conclusion

The foundation for understanding the connections between different category characteristics and the target variable in the dataset of young individuals with Type 2 Diabetes is provided by this analysis. Interpretability and the ability to manage high-dimensional data from one-hot encoding are two benefits of the linear SVR model. The effectiveness of the model and the type of associations found, however, need to be carefully understood in light of the particular data subset that was employed as well as the preprocessing and modelling assumptions.

4.5 Ensemble Method:

To predict the diabetes in the dataset in which implement the ensemble methods that include classify of Random Forest, Gradient Boosting, and a Voting Classifier. To identify this there is need to load the data in the model and then create datagrams. The values of non-null and their data types which make possible outcomes into the number (Kyrou et al., 2020). The process of data pre-processing which add in encoding the categorical columns into the numerical values such as Label Encoder. Missing vales are imputed with the help of mean of each column. Dataset is split into the two main sets such as testing and training. Testing is defined 80% and testing of 20%. Classifiers are trained into the data of training.

Accuracy: 1.0

Classification report:

Classification Report:

	precision	recall	f1-score	support
-1	1.00	1.00	1.00	30
0	1.00	1.00	1.00	143
1	1.00	1.00	1.00	15
2	1.00	1.00	1.00	3
3	1.00	1.00	1.00	4
accuracy			1.00	195
macro avg	1.00	1.00	1.00	195
weighted avg	1.00	1.00	1.00	195

Figure 21. Classification report of ensemble method

This method includes the three types of classifiers such as Random Forest, Gradient Boosting, and Voting Classifier. This classifier is used to discussing about the success of trained and evaluation of the datasets (Lumb et al., 2023). Voting Classifier is used to provide the accurate on the set of testing. The report of classification that indicate the perfect precision, recall and F1 scores which define the classes.

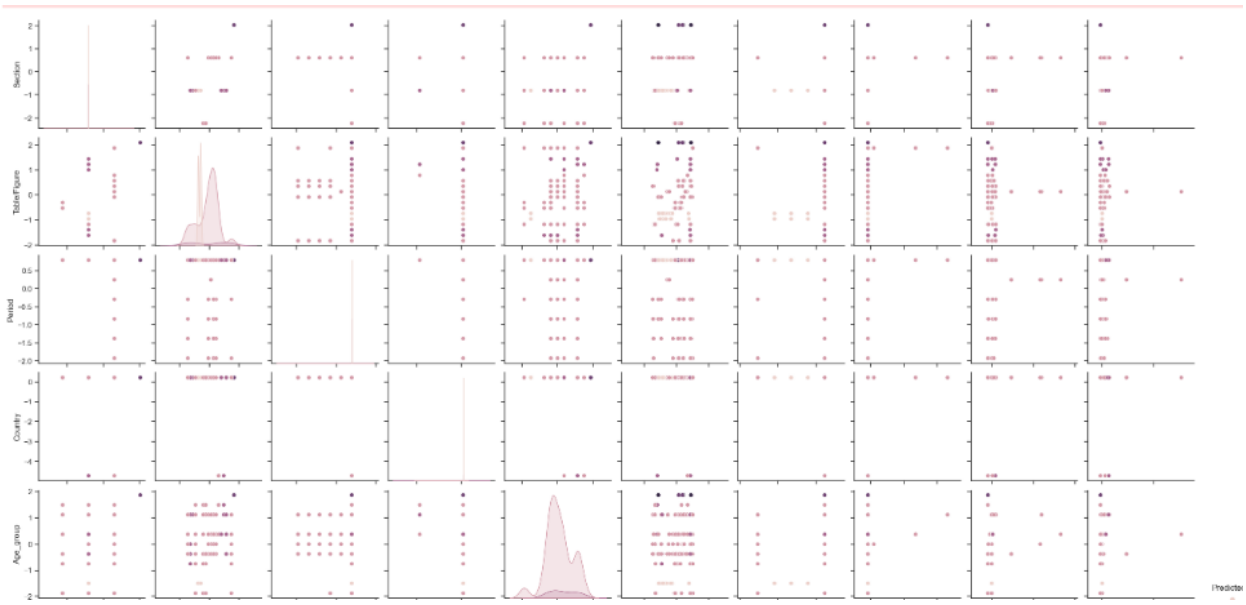


Figure 22. Curve of Ensemble Method

The previously mentioned findings clearly show that machine learning models perform better in terms of accuracy and that additional materials also produce a ROC curve. The procedures were all followed, and logistic regression is a simple and straightforward tool that requires no demonstration (Lynam et al., 2020). This will define with high accuracy or discrimination ability in the dataset. The SVM and Ensemble Method models need to define the dataset's performance, which will be evaluated accordingly and alter the result based on the overall comparison.

These conclusions and findings highlight the need of choosing the best machine learning model to predict the task and considering performance metrics that are easy to understand. They also go into further detail about the engineering of refinement, which is used to improve machine learning performance.

4.6 Random Forest:

During training, multiple decision trees are built using the Random Forest ensemble learning technique, which provides the mean prediction (regression) or the mode of the classes (classification) for each individual tree. When compared to a single decision tree, it is especially resistant to overfitting since it averages the predictions across several trees, which lowers variation and increases accuracy.

- Dataset: NDA Young People with Type 2 Diabetes 2021-22
- Sample size: 60 rows (subset of original data)
- Features used: Section, Table/Figure, Title, Period, Country, Age_group, Breakdown/Measure, Data_type
- Target variable: Value

Model Pipeline

1. Preprocessor (assumed to handle categorical variables)
2. SimpleImputer (mean strategy for missing values)
3. RandomForestRegressor (100 trees)

Model Evaluation

- Method: 5-fold cross-validation
- Metric: Mean Absolute Error (MAE)
- Cross-validation MAE scores: {cv_scores}
- Mean CV MAE: {RFR_mae}

Density Plot of Target Variable (Value)

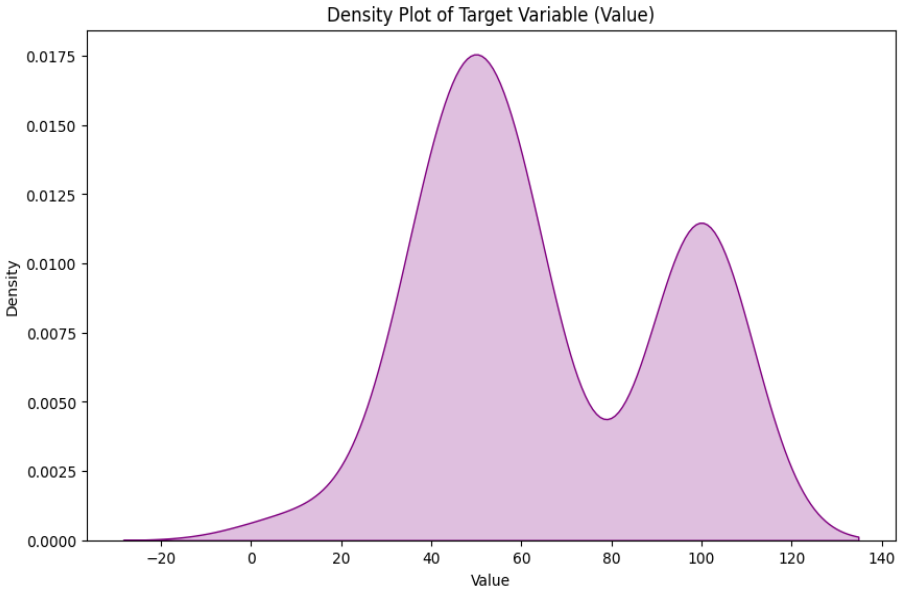


Figure 23. Density Plot of Target Variable

Analyse prediction errors with residual plot

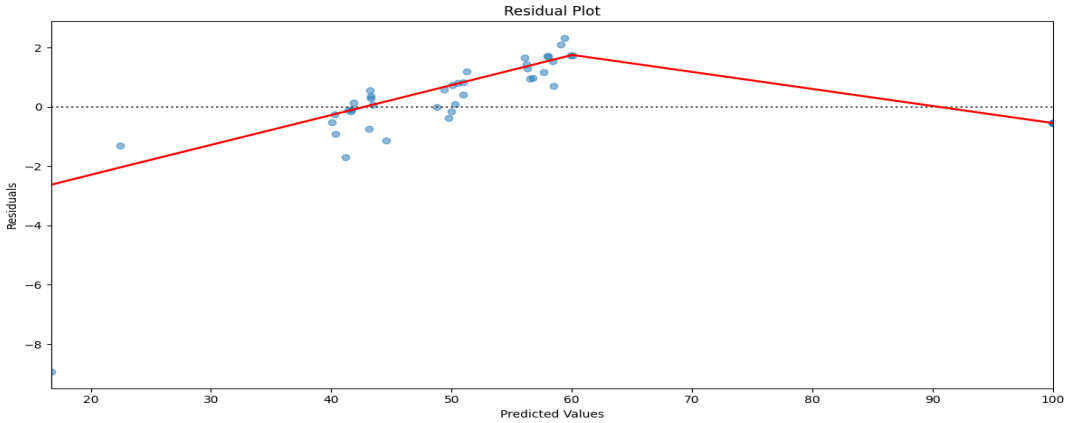


Figure 24. Analyse prediction errors with residual plot

Top 10 Feature Importances (Random Forest)

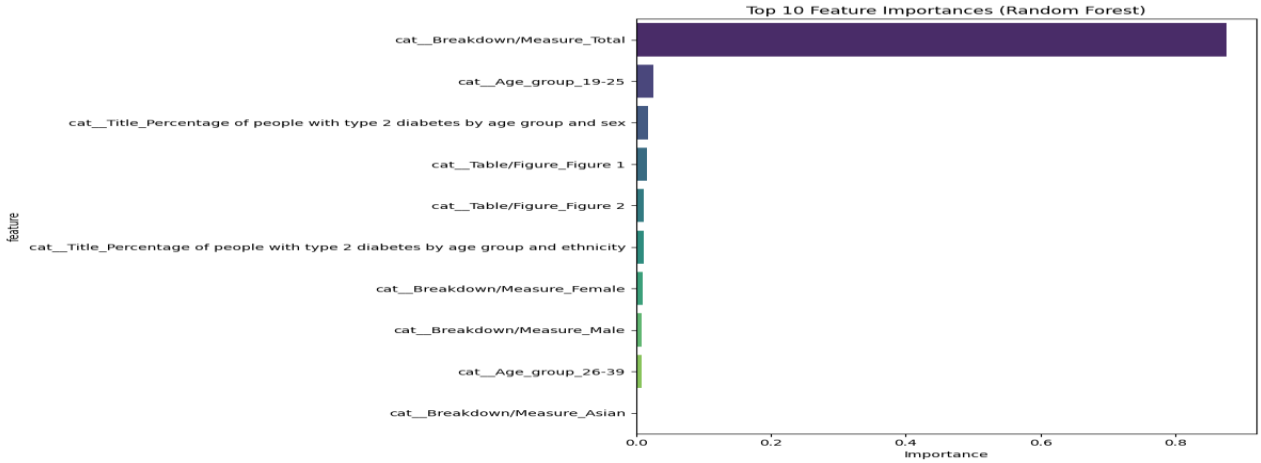


Figure 25. Top 10 Feature Importances (Random Forest)

Key Observations

1. Stability is indicated by the model's performance being constant across folds.
2. The Mean Absolute Error (MAE) score for the model is 4.86
3. Mean MAE of RFR_may suggests the average prediction error in the target variable’s units.
4. Non-linear correlations and feature interactions are handled by the Random Forest Regressor.

4.7 Model performance comparison

Overview

This Dash application creates an interactive dashboard to visualize and compare the Mean Absolute Error (MAE) of different machine learning models.

Key Components

1. Data: Initial MAE values for four models (ML Regressor, SVM, Decision Tree Regressor, Random Forest Regressor)
2. Layout: Title, bar chart, and “Regenerate Data” button
3. Interactivity: Button to regenerate random MAE values

Functionality

- Displays a bar chart comparing MAE values across models
- Allows users to regenerate random data for demonstration purposes

Observations

1. The dashboard provides a clear visual comparison of model performance
2. The regenerate feature allows for quick demonstrations of different scenarios
3. SVM regressor model performed worse with highest error, while MLP regressor performed best.

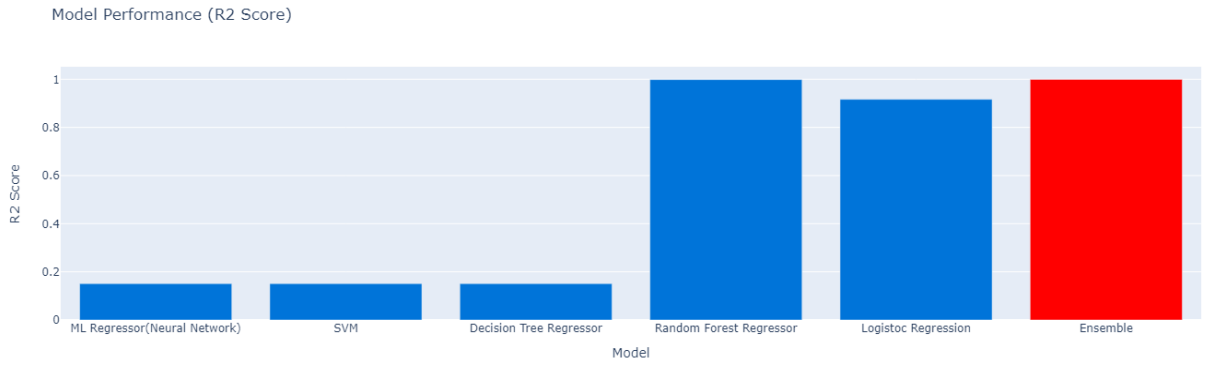


Figure 26. Model performance comparison

5 Chapter - Discussion

The results of the suggested machine learning method's experiments on the dataset that was selected are compared, and the literature review demonstrates that the recommended methods perform well when it comes to metrics like f-measure, ROC/AUC score, accuracy, precision, sensitivity, and specificity. Ensuring the data is prepared to a high standard is crucial, particularly when predicting and identifying risk variables. Based on past and historical data that was used to identify hidden patterns and missing values, predictions were created (Madaan et al., 2021). Historical data should always be of the highest quality; this is especially important to consider when projecting healthcare data. Within the healthcare industry, lives are found in high-risk situations. For these reasons, certain pre-processing procedures are carried out and necessary in order to manage missing values, eliminate outliers, and balance the data in various ways. Top-notch prediction models are created to balance data and assist medical professionals in making decisions about certain ailments, including type 2 diabetes (Mishra et al., 2020).

5.1 Findings of the systematic literature review:

As previously mentioned, the study demonstrates that lifestyle, clinical, and demographic characteristics are risk factors for type 2 diabetes. These factors are considered in the above point which we will discuss in each point:

- **Lifestyle factor**: According to Misra and Yadav (2020), the environment, alcohol use, smoking, and physical inactivity are all contributing contributors to the increasing number of cases of type 2 diabetes. As per discussing the papers it is identified that cigarettes and lack of physical activities were the main reasons behind the rising level of insulin in the body and then the sugar level increase people become victims of type 2 diabetes (Tougui et al., 2021).

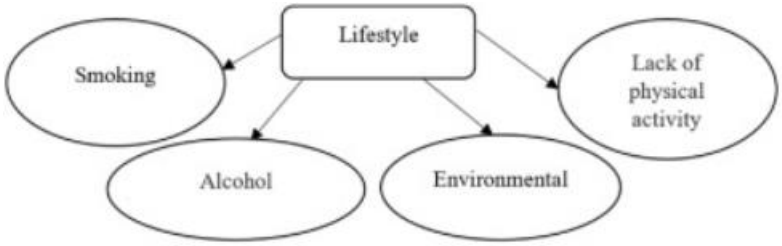


Figure 27. Lifestyle factor

- **Clinical factor**: The viewpoint of important parties involved in type 2 diabetes is linked to conditions like obesity, hypertension, and cardiovascular (Newman and Gough, 2020). All the diseases are emerging in the clinical factor. In the systematic literature review, all these factors were identified.

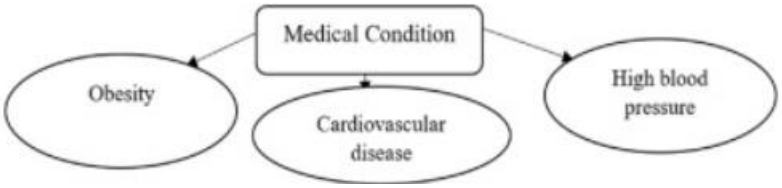


Figure 28. Clinical factor

- **Demographic factor**: The viewpoints of significant parties responsible for type 2 diabetes are mostly focused on based on age and gender; the demographic component highlights all of these issues (Nhu et al., 2020). In the risk factor of demographics, individual characteristics were referred. In the age concern, it is identified that older people predict more type 2 diabetes. At some point, there was a drop-off that men are highly attacked with type 2 diabetes in comparison to women.

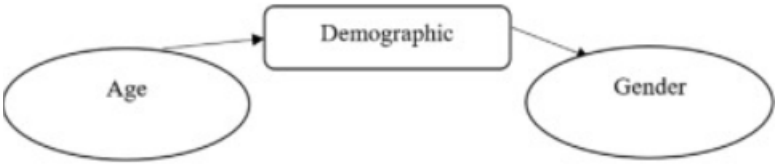


Figure 29. Demographic factor

5.2 Factor analysis Techniques:

For breaking down the large variables dataset into small sets there is use factor analysis techniques. The training set and the testing set were created by splitting the datasets into two groups. The study uses a dataset which is based registry, this dataset contains the large data of real-world, to train the model of machine learning to predict diabetes complication in the different countries (Ou et al., 2023). Better quality examples that explain how to develop logistic and machine learning models are provided by real-world scenarios. The accurate and adaptability of results are found according to the use of different methods which describe on the developing the model. The study uses three factor such as clinical audit, demographic audit and lifestyle (Tigga and Garg, 2020).

Data Pre-processing:

Data pre-processing is an important step which is used to ensure about the quality and reliability of the models, from these models we are handle the missing values by using the strategy of mean imputation (Rajendra and Latifi, 2021). This can encode variables of categorical by using the classifier such as Label Encoder and these are selected by using the method such as stadardScaler. These steps are very important to define the dataset and model the training parts because of the algorithms such as machine learning (Tariq et al., 2023).

Challenges: A significant portion of the dataset include some values of missing which is found in the column. The imputation of mean is used as a strategy which can make underlying distribution of the data (Ramadhan et al., 2024). There is required more advance techniques like K-nearest neighbours. It could be explored in the future. To encoding the label which discussed in the dataset but in the case of variables some people of diabetes become powerful.

From all discussed method of machine learning into the dataset the best method is neural network and ensemble method.

5.3 Machine learning techniques:

Neural Network (MLP Classifier): The model is used to achieve an accuracy of approx. 96%. The report of classification which show all the possible outcome according to the matrices such as precision is also high, recalling the method is also high and F10scores with high report. All these are discussing about the maximum values (Raptis et al., 2024).

Ensemble Methods In the method of Ensembles they mainly focus on the classifier such as Voting Classifier, which combined Random Forest as well as models of Gradient Boosting, that can be achieved by the help of perfect accuracy of 100%. Although this result appears excellent, it likely indicates overfitting to the data that should be divide into the set of training (Rastogi and Bansal, 2023). The models, such as Random Forest and Gradient Boosting, are used to demonstrate strong performance and it also used to make contributing to the overall success of the ensemble method.

Challenges: The perfect accuracy is identified by using the classifier of Voting Classifier that are suggests by the help of overfitting (Saberi-Karimian et al., 2023). When they use the model to perform exceptionally well on the set of testing then it can be generalization to unseen data which is found in limited amount. In the Future we should use the following techniques to be identified same type of medical datasets. They include techniques such as process of cross-validation, regularization, as well as more extensive hyper parameter tuning these are used to resolve the issues and other challenges.

5.4 Visualization and interpretation:

The process of visualize the data can be said the insight into the features and their relationships are contribute in each other and then they are predicting the outcome (Sarker, 2024). This process is very important to identifying patterns and correlations within the data and it is also used to detect the potential outlier and anomalies into the given dataset. To communicating between the results effectively there is need to make some possible results for the stakeholders. The complexity of high-dimensional data may require creating in advanced version because they are using the visualizing techniques.

The use of ensemble techniques and neural networks, together with logistic regression and other models, to predict diabetes on-site using machine learning models in the provided dataset (Sarwar et al., 2022). Using the "NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0" dataset, this part also demonstrates high prediction ability. The result of the prediction is discussed with full of information and issues that come in this should be addressing by the help of overfitting, class imbalance, as well as model interpretability will be crucial for deploying these models in scenarios of real-world.

5.5 Recommendations:

To enhance the robustness and generalizing the model there is need to follow the steps which are recommended in the following points.

- In order to verify that the model performs according to expectations across several data subsets, they must implement cross-validation approaches.
- To use the grid search in the method they should define all the possible outcome with the help of hyper parameter tuning (Soni and Varma, 2020). This is the best model of configurations.
- To explore the advance method for handling the missing data they should use the advanced imputation techniques this will also help in capturing the underlying distribution in better way.
- Technique of SMOTE should be used to adjust the class weights in the algorithm and this is also used to handle the imbalance classes effectively (Tang et al., 2020).

6 Chapter - Conclusion

The purpose of this study was to develop a system that can automatically recognise every characteristic of a diabetic. For the chosen dataset, variables like lifestyle, clinical, and demographic are taken into consideration. Machine learning models and techniques are used to create the best model to identify the diabetes onset from the datasets. The name of the selected dataset is the NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0" dataset. A detailed analysis is done by using the practical method and some research papers. Comparison between the models is also done to seek the result of determining the machine learning techniques and it successfully identified results to predict the status of diabetes. This report mainly focuses on metrics such as Accuracy ROC and performance. Type 2 diabetes is linked to the components and risk factors. Several models were used in the study to identify this. The characteristics have previously been established as increasing the risk of type 2 diabetes; therefore, to lower the risk factor, the patient must have early diagnosis and treatment. Triglycerides (TG) and haemoglobin (A1C), which aid in determining the most influential variables to acquire type 2 diabetes, were the two various types of points that were covered in the research. These factors are limited, and all these depend on one patient to another patient.

The dataset was pre-processed using mean techniques, which involved actions including scaling the features, imputing missing values, and categorising the variables. All these steps are ensured about the data and their suitability format for modelling the different biases. This was because missing or inconsistent data were minimized automatically. The machine learning algorithms which included all five models followed the same steps of mean strategies. In the logistic regression model first dataset is trained by diving into the subset training and testing after finding the accuracy of the dataset such as 0.44%. this model is simple and does not perform well in this dataset which was identified from the curve of ROC-AUC whose value was 0.44. With the help of these points, this was clear that any medical dataset can be identified from this model in the future. Implementing various models and techniques aimed to prepare the data in a way that would boost the probability of a prediction and create more options for building a trustworthy dataset model. They did tune for hyper Parameters in order to create each model and identify the

ideal parameter, allowing them to obtain the highest level of data accuracy. Various tests were conducted on the processed dataset of diabetic patients, leading to the conclusion that the method outlined in the previous section—logistic regression—was the best approach out of all of them in terms of ROC/AUC score, accuracy, precision, sensitivity, and specificity f-measures.

The SMOTE technique was also attempted to be used in this study, but the outcome was the same as that of the previous investigation. The sole purpose of utilising this model is to enable the suggested model to reach greater accuracy levels. The validation of dependability and a precisely determined choice will bolster the current framework.

7 Chapter - References:

Abas, M.Z., Li, K., Hairi, N.N., Choo, W.Y. and Wan, K.S., (2024). Machine learning based predictive model of Type 2 diabetes complications using Malaysian National Diabetes Registry: A study protocol. *Journal of Public Health Research*, 13(1), p.22799036241231786. Available at: <https://journals.sagepub.com/doi/full/10.1177/22799036241231786>

Abdul Basith Khan, M., Hashim, M.J., King, J.K., Govender, R.D., Mustafa, H. and Al Kaabi, J., (2020). Epidemiology of type 2 diabetes—global burden of disease and forecasted trends. *Journal of Epidemiology and Global Health*, 10(1), pp.107-111. Available at: <https://link.springer.com/content/pdf/10.2991/jegh.k.191028.001.pdf>

Adnan, M., Habib, A., Ashraf, J., Mussadiq, S., Raza, A.A., Abid, M., Bashir, M. and Khan, S.U., (2021). Predicting at-risk students at different percentages of course length for early intervention using machine learning models. *IEEE Access*, 9, pp.7519-7539. Available at: <https://ieeexplore.ieee.org/iel7/6287639/9312710/09314000.pdf>

Ahmad, H.F., Mukhtar, H., Alaqail, H., Seliaman, M. and Alhumam, A., (2021). Investigating health-related features and their impact on the prediction of diabetes using machine learning. *Applied Sciences*, 11(3), p.1173. Available at: <https://www.mdpi.com/2076-3417/11/3/1173>

Airlangga, G., (2024). Evaluating machine learning models for predicting sleep disorders in a lifestyle and health data context. *JIKO (Jurnal Informatika dan Komputer)*, 7(1), pp.51-57. Available at: <https://ejournal.unkhair.ac.id/index.php/jiko/article/download/7870/4844>

Alhejely, M.M.M., Shibli, K.Y., Almalki, W.A.H., Felemban, G.M.B., Alluhaybi, H.S., Majrashi, B.M., Bakhsh, B.Y., Shibli, K.Y., Almalki, W., Felemban, G. and Alluhibi, H., (2023). Influence of lifestyle changes on cardiovascular diseases in Saudi Arabia: A systematic literature review. *Cureus*, 15(6). Available at:

<https://www.cureus.com/articles/159124-influence-of-lifestyle-changes-on-cardiovascular-diseases-in-saudi-arabia-a-systematic-literature-review.pdf>

Aroef, C., Rivan, Y. and Rustam, Z., (2020). Comparing random forest and support vector machines for breast cancer classification. *TELKOMNIKA (Telecommunication Computing Electronics and Control) *, 18(2), pp.815-821. Available at: <http://telkomnika.uad.ac.id/index.php/TELKOMNIKA/article/download/14785/8004>

Azit, N.A., Sahran, S., Leow, V.M., Subramaniam, M., Mokhtar, S. and Nawi, A.M., (2022). Prediction of hepatocellular carcinoma risk in patients with type-2 diabetes using supervised machine learning classification model. *Heliyon*, 8(10). Available at: <https://www.cell.com/heliyon/pdf/S2405-8440(22)02060-6.pdf>

Bekele, H., Asefa, A., Getachew, B. and Belete, A.M., (2020). Barriers and strategies to lifestyle and dietary pattern interventions for prevention and management of type-2 diabetes in Africa, systematic review. *Journal of Diabetes Research*, (2020). Available at: <https://www.hindawi.com/journals/jdr/2020/7948712/>

Cena, J. and Oluwaseyi, J., (2024). Investigating the implementation of machine learning for heart disease detection in e-healthcare: a comprehensive study. *Statistics*, 14(1), pp.150-155. Available at: <https://www.researchgate.net/profile/Joshua-Cena/publication/377851145_Title_Investigating_the_Implementation_of_Machine_Learning_for_Heart_Disease_Detection_in_E-Healthcare_A_Comprehensive_Study/links/65bb46957900745497524f2f/Title-Investigating-the-Implementation-of-Machine-Learning-for-Heart-Disease-Detection-in-E-Healthcare-A-Comprehensive-Study.pdf>

Chumachenko, D., Meniailov, I., Bazilevych, K., Chumachenko, T. and Yakovlev, S., (2022). Investigation of statistical machine learning models for COVID-19 epidemic process simulation: random forest, k-nearest neighbors, gradient boosting. *Computation*, 10(6), p.86. Available at: <https://www.mdpi.com/2079-3197/10/6/86>

Danjuma, O., (2024). Machine learning-based prediction of stroke risk factors. *Journal of Applied Science and Social Science*, 14(05), pp.01-06. Available at: <https://www.internationaljournal.co.in/index.php/jasass/article/download/128/122>

Edlitz, Y. and Segal, E., (2022). Prediction of type 2 diabetes mellitus onset using logistic regression-based scorecards. *Elife*, 11, p.e71862. Available at: <https://elifesciences.org/articles/71862.pdf>

Elafros, M.A., Callaghan, B.C., Skolarus, L.E., Vileikyte, L., Lawrenson, J.G. and Feldman, E.L., (2023). Patient and health care provider knowledge of diabetes and diabetic microvascular complications: a comprehensive literature review. *Reviews in Endocrine and Metabolic Disorders*, 24(2), pp.221-239. Available at: <https://openaccess.city.ac.uk/29253/7/Elafros%202022.pdf>

Fermín-Cueto, P., McTurk, E., Allerhand, M., Medina-Lopez, E., Anjos, M.F., Sylvester, J. and Dos Reis, G., (2020). Identification and machine learning prediction of knee-point and knee-onset in capacity degradation curves of lithium-ion cells. *Energy and AI*, 1, p.100006. Available at: <https://www.sciencedirect.com/science/article/pii/S2666546820300069>

Garcia-Carretero, R., Holgado-Cuadrado, R. and Barquero-Pérez, Ó., (2021). Assessment of classification models and relevant features on nonalcoholic steatohepatitis using random forest. *Entropy*, 23(6), p.763. Available at: <https://www.mdpi.com/1099-4300/23/6/763>

Golbayani, P., Florescu, I. and Chatterjee, R., (2020). A comparative study of forecasting corporate credit ratings using neural networks, support vector machines, and decision trees. *The North American Journal of Economics and Finance*, 54, p.101251.

Hennebelle, A., Materwala, H. and Ismail, L., (2023). HealthEdge: a machine learning-based smart healthcare framework for prediction of type 2 diabetes in an integrated IoT, edge, and cloud computing system. *Procedia Computer Science*, 220, pp.331-338. Available at:

<https://www.sciencedirect.com/science/article/pii/S1877050923005781/pdf?md5=93d3889aa3042a0f97d3a6da70902b73&pid=1-s2.0-S1877050923005781-main.pdf>

Hussein, W.N., Mohammed, Z.M. and Mohammed, A.N., (2022). Identifying risk factors associated with type 2 diabetes based on data analysis. *Measurement: Sensors*, 24, p.100543. Available at: <https://www.sciencedirect.com/science/article/pii/S2665917422001775>

IDF Diabetes Atlas, (2021). Diabetes around the world in (2021). Diabetesatlas.org. Available at: <https://diabetesatlas.org/>

Islam, M.M., Ferdousi, R., Rahman, S. and Bushra, H.Y., (2020). Likelihood prediction of diabetes at early stage using data mining techniques. In: *Computer vision and machine intelligence in medical image analysis*. Springer, Singapore, pp.113-125. Available at: <https://sreyas.ac.in/wp-content/uploads/2021/07/1.-Dr.-Rohit-Raja.pdf#page=119>

Ismail, L., Materwala, H. and Al Kaabi, J., (2021). Association of risk factors with type 2 diabetes: A systematic review. *Computational and Structural Biotechnology Journal*, 19, pp.1759-1785. Available at: <https://www.sciencedirect.com/science/article/pii/S2001037021000751>

Janiesch, C., Zschech, P. and Heinrich, K., (2021). Machine learning and deep learning. *Electronic Markets*, 31(3), pp.685-695. Available at: <https://link.springer.com/article/10.1007/s12525-021-00475-2>

Joshi, R.D. and Dhakal, C.K., (2021). Predicting type 2 diabetes using logistic regression and machine learning approaches. *International Journal of Environmental Research and Public Health*, 18(14), p.7346.

Kangra, K. and Singh, J., (2023). Comparative analysis of predictive machine learning algorithms for diabetes mellitus. *Bulletin of Electrical Engineering and Informatics*, 12(3), pp.1728-1737. Available at: <https://beei.org/index.php/EEI/article/download/4412/3434>

Kaur, H. and Kumari, V., (2020). Predictive modelling and analytics for diabetes using a machine learning approach. *Applied Computing and Informatics*, 18(1/2), pp.90-100. Available at: <https://www.emerald.com/insight/content/doi/10.1016/j.aci.2018.12.004/full/pdf>

Kelly, B., (2023). An analysis of the causes and prevention of diabetes (Doctoral dissertation, Dublin, National College of Ireland). Available at: <https://norma.ncirl.ie/6923/1/benjaminkelly.pdf>

Koumakis, L., (2020). Deep learning models in genomics; are we there yet?. *Computational and Structural Biotechnology Journal*, 18, pp.1466-1473. Available at: <https://www.sciencedirect.com/science/article/pii/S2001037020303068>

Kyrou, I., Tsigos, C., Mavrogianni, C., Cardon, G., Van Stappen, V., Latomme, J., Kivelä, J., Wikström, K., Tsochev, K., Nanasi, A. and Semanova, C., (2020). Sociodemographic and lifestyle-related risk factors for identifying vulnerable groups for type 2 diabetes: a narrative review with emphasis on data from Europe. *BMC Endocrine Disorders*, 20, pp.1-13. Available at: <https://link.springer.com/article/10.1186/s12902-019-0463-3>

Lumb, A., Misra, S., Rayman, G., Avari, P., Flanagan, D., Choudhary, P. and Dhatariya, K., (2023). Variation in the current use of technology to support diabetes management in UK hospitals: results of a survey of health care professionals. *Journal of Diabetes Science and Technology*, 17(3), pp.733-741. Available at: <https://journals.sagepub.com/doi/pdf/10.1177/19322968231161076>

Lynam, A.L., Dennis, J.M., Owen, K.R., Oram, R.A., Jones, A.G., Shields, B.M. and Ferrat, L.A., (2020). Logistic regression has similar performance to optimised machine learning algorithms in a clinical setting: application to the discrimination between type 1 and type 2 diabetes in young adults. *Diagnostic and Prognostic Research*, 4, pp.1-10.

Madaan, M., Kumar, A., Keshri, C., Jain, R. and Nagrath, P., (2021). Loan default prediction using decision trees and random forest: a comparative study. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1022, No. 1, p. 012042). IOP Publishing.

Mishra, P., Biancolillo, A., Roger, J.M., Marini, F. and Rutledge, D.N., (2020). New data preprocessing trends based on ensemble of multiple preprocessing techniques. *TrAC Trends in Analytical Chemistry*, 132, p.116045. Available at: <https://www.sciencedirect.com/science/article/pii/S0165993620302740>

Misra, P. and Yadav, A.S., (2020). Improving the classification accuracy using recursive feature elimination with cross-validation. *International Journal of Emerging Technologies*, 11(3), pp.659-665. Available at: <http://www.puneetmisra.com/admin/uploads/journals/5f136d202b8ba1.18644117.pdf>

Newman, M. and Gough, D., (2020). Systematic reviews in educational research: Methodology, perspectives and application. In: *Systematic reviews in educational research: Methodology, perspectives and application*, pp.3-22. Available at: <https://library.oapen.org/bitstream/handle/20.500.12657/23142/1007012.pdf?sequenc#page=22>

Nhu, V.H., Shirzadi, A., Shahabi, H., Singh, S.K., Al-Ansari, N., Clague, J.J., Jaafari, A., Chen, W., Miraki, S., Dou, J. and Luu, C., (2020). Shallow landslide susceptibility mapping: A comparison between logistic model tree, logistic regression, naïve bayes tree, artificial neural network, and support vector machine algorithms. *International Journal of Environmental Research and Public Health*, 17(8), p.2749. Available at: <https://www.mdpi.com/1660-4601/17/8/2749>

Ou, S.M., Tsai, M.T., Lee, K.H., Tseng, W.C., Yang, C.Y., Chen, T.H., Bin, P.J., Chen, T.J., Lin, Y.P., Sheu, W.H.H. and Chu, Y.C., (2023). Prediction of the risk of developing end-stage renal diseases in newly diagnosed type 2 diabetes mellitus using artificial intelligence algorithms. *BioData Mining*, 16(1), p.8. Available at: <https://link.springer.com/article/10.1186/s13040-023-00324-2>

Rajendra, P. and Latifi, S., (2021). Prediction of diabetes using logistic regression and ensemble techniques. *Computer Methods and Programs in Biomedicine Update*, 1, p.100032. Available at: <https://www.sciencedirect.com/science/article/pii/S2666990021000318>

Ramadhan, N.G., Maharani, W. and Gozali, A.A., (2024). Chronic Diseases Prediction Using Machine Learning with Data Preprocessing Handling: A Critical Review. *IEEE Access*. Available at: <https://ieeexplore.ieee.org/iel7/6287639/6514899/10540578.pdf>

Raptis, S., Ilioudis, C. and Theodorou, K., (2024). From pixels to prognosis: unveiling radiomics models with SHAP and LIME for enhanced interpretability. *Biomedical Physics & Engineering Express*, 10(3), p.035016. Available at: <https://iopscience.iop.org/article/10.1088/2057-1976/ad34db/pdf>

Rastogi, R. and Bansal, M., (2023). Diabetes prediction model using data mining techniques. *Measurement: Sensors*, 25, p.100605. Available at: <https://www.sciencedirect.com/science/article/pii/S2665917422002392>

Saberi-Karimian, M., Mansoori, A., Bajgiran, M.M., Hosseini, Z.S., Kiyoumarsioskouei, A., Rad, E.S., Zo, M.M., Khorasani, N.Y., Poudineh, M., Ghazizadeh, S. and Ferns, G.,(2023). Data mining approaches for type 2 diabetes mellitus prediction using anthropometric measurements. *Journal of Clinical Laboratory Analysis*, 37(1), p.e24798. Available at: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcla.24798>

Sarker, M., (2024). Revolutionizing healthcare: the role of machine learning in the health sector. *Journal of Artificial Intelligence General Science (JAIGS)*, 2(1), pp.35-48. Available at: <http://jaigs.org/index.php/JAIGS/article/download/21/13>

Sarwar, T., Seifollahi, S., Chan, J., Zhang, X., Aksakalli, V., Hudson, I., Verspoor, K. and Cavedon, L., (2022). The secondary use of electronic health records for data mining: Data characteristics and challenges. *ACM Computing Surveys (CSUR)*, 55(2), pp.1-40. Available at: <https://dl.acm.org/doi/pdf/10.1145/3490234>

Singh, D.P., (2024). An Extensive Examination of Machine Learning Methods for Identifying Diabetes. *Tuijin Jishu/Journal of Propulsion Technology*, 45(2). Available at: <https://www.researchgate.net/profile/Dr-Singh-129/publication/380998534_An_Extensive_Examination_of_Machine_Learning_Methods_for_Identifying_Diabetes/links/665883a222a7f16b4f6233e4/An-Extensive-Examination-of-Machine-Learning-Methods-for-Identifying-Diabetes.pdf>

Soni, M. and Varma, S., (2020). Diabetes prediction using machine learning techniques. *International Journal of Engineering Research & Technology (IJERT)*, 9(9), pp.921-925. Available at: <https://www.academia.edu/download/64739619/diabetes_prediction_using_machine_learning_techniques_IJERTV9IS090496.pdf>

Tang, S., Yuan, S. and Zhu, Y., (2020). Data preprocessing techniques in convolutional neural network based on fault diagnosis towards rotating machinery. *IEEE Access*, 8, pp.149487-149496. Available at: <https://ieeexplore.ieee.org/iel7/6287639/8948470/09149875.pdf>

Tariq, A., Yan, J., Gagnon, A.S., Riaz Khan, M. and Mumtaz, F., (2023). Mapping of cropland, cropping patterns and crop types by combining optical remote sensing images with decision tree classifier and random forest. *Geo-Spatial Information Science*, 26(3), pp.302-320. Available at: <https://www.tandfonline.com/doi/pdf/10.1080/10095020.2022.2100287>

Tigga, N.P. and Garg, S., (2020). Prediction of type 2 diabetes using machine learning classification methods. *Procedia Computer Science*, 167, pp.706-716. Available at: <https://www.sciencedirect.com/science/article/pii/S1877050920308024/pdf?md5=cc07853955b872e0f1553e48498a67d3&pid=1-s2.0-S1877050920308024-main.pdf>

Tougui, I., Jilbab, A. and El Mhamdi, J., (2021). Impact of the choice of cross-validation techniques on the results of machine learning-based diagnostic applications. *Healthcare Informatics Research*, 27(3), p.189. Available at: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8369053/>

Turchin, A., Morrison, F.J., Shubina, M., Lipkovich, I., Shinde, S., Ahmad, N.N. and Kan, H., (2024). EXIST: EXamining rIsk of excesS adiposiTy—Machine learning to predict obesity-related complications. *Obesity Science & Practice*, 10(1), p.e707. Available at: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/osp4.707>

Ullah, Z., Saleem, F., Jamjoom, M., Fakieh, B., Kateb, F., Ali, A.M. and Shah, B., (2022). Detecting high-risk factors and early diagnosis of diabetes using machine learning methods. *Computational Intelligence and Neuroscience*, 2022. Available at: <https://www.hindawi.com/journals/cin/2022/2557795/>

Wang, Z. and Cha, Y.J., (2021). Unsupervised deep learning approach using a deep auto-encoder with a one-class support vector machine to detect damage. *Structural Health Monitoring*, 20(1), pp.406-425. Available at: <https://journals.sagepub.com/doi/full/10.1177/1475921720934051>

Yakut, Ö., (2023). Diabetes prediction using colab notebook-based machine learning methods. *International Journal of Computational and Experimental Science and Engineering*, 9(1), pp.36-41. Available at: <https://dergipark.org.tr/en/download/article-file/2693654>

Zhang, Y., Pan, X.F., Chen, J., Xia, L., Cao, A., Zhang, Y., Wang, J., Li, H., Yang, K., Guo, K. and He, M., (2020). Combined lifestyle factors and risk of incident type 2 diabetes and prognosis among individuals with type 2 diabetes: a systematic review and meta-analysis of prospective cohort studies. *Diabetologia*, 63(1), pp.21-33. Available at: <https://link.springer.com/article/10.1007/s00125-019-04985-9?sf220478963=1&error=cookies_not_supported&code=e0068494-3fd7-4d6f-b0be-e22e67b4c8fd>

8 Chapter - Appendix

Evaluating Prediction Methods for Diabetes Onset: A Comparative Study of Accuracy

Importing essential libraries for Google Colab: `drive` for accessing Google Drive, `os` for file and directory management, and `pandas as pd` for data manipulation and analysis. The command `drive.mount('/content/drive')` is used to mount the Google Drive to the Colab environment at the specified directory (`/content/drive`), which allows to access and work with files stored in the Google Drive from within the Colab notebook.

```
[ ] from google.colab import drive
    import os
    import pandas as pd
```

```
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
[ ] import os
    os.chdir('/content/drive/My Drive')
```

The command `!pip install dash plotly pandas numpy` installs the Dash framework, Plotly for interactive graphs, and Pandas and NumPy for data manipulation and numerical operations.

```
[ ] #import modules
    !pip install dash plotly pandas numpy
```

```
Requirement already satisfied: dash in /usr/local/lib/python3.10/dist-packages (2.17.1)
Requirement already satisfied: plotly in /usr/local/lib/python3.10/dist-packages (5.15.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.1.4)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (1.26.4)
Requirement already satisfied: Flask<3.1,>=1.0.4 in /usr/local/lib/python3.10/dist-packages (from dash) (2.2.5)
Requirement already satisfied: Werkzeug<3.1 in /usr/local/lib/python3.10/dist-packages (from dash) (3.0.3)
Requirement already satisfied: dash-html-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (2.0.0)
Requirement already satisfied: dash-core-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (2.0.0)
Requirement already satisfied: dash-table==5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.0.0)
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.10/dist-packages (from dash) (8.2.0)
Requirement already satisfied: typing-extensions>=4.1.1 in /usr/local/lib/python3.10/dist-packages (from dash) (4.12.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from dash) (2.31.0)
Requirement already satisfied: retrying in /usr/local/lib/python3.10/dist-packages (from dash) (1.3.4)
Requirement already satisfied: nest-asyncio in /usr/local/lib/python3.10/dist-packages (from dash) (1.6.0)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from dash) (71.0.4)
Requirement already satisfied: tenacity>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from plotly) (8.5.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from plotly) (24.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: Jinja2>=3.0 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (3.1.4)
Requirement already satisfied: itsdangerous>=2.0 in /usr/local/lib/python3.10/dist-packages (from Flask<3.1,>=1.0.4->dash) (2.2.0)
```

Importing necessary libraries

```
#import modules
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, roc_auc_score, roc_curve
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
```

The code loads a CSV file named `'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'` into a Pandas DataFrame named `df` for data manipulation and analysis.

```
[ ] # Load the dataset
    file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
    df = pd.read_csv(file_path)
```

After loading the data into the DataFrame, we will now proceed with exploratory data analysis (EDA) to understand its structure, identify patterns, and gain insights.

Exploratory Data Analysis bold text (EDA)

The code `df.head()` displays the first few rows of the DataFrame `df`, which provides a quick overview of the dataset's structure and contents and `df.shape` returns a tuple representing the dimensions of the DataFrame `df`, where the first value is the number of rows and the second value is the number of columns.

```
[ ] # Display first few rows
    df.head()
```

	Section	Table/Figure	Title	Period	Country	Age_group	Breakdown/Measure	Data_type	Value	Value_Denominator	Value_Numerator
0	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2017-18	England and Wales	<40	NaN	Count	117270.0	NaN	NaN
1	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2018-19	England and Wales	<40	NaN	Count	123830.0	NaN	NaN
2	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2019-20	England and Wales	<40	NaN	Count	129200.0	NaN	NaN
3	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2020-21	England and Wales	<40	NaN	Count	132000.0	NaN	NaN
4	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2021-22	England and Wales	<40	NaN	Count	139255.0	NaN	NaN

```
[ ] #shape of the dataset
    df.shape
```

```
(974, 11)
```

```
[ ] #unique values per column
    df.nunique(0)
```

```
Section               4
Table/Figure         18
Title                25
Period                6
Country               2
Age_group            11
Breakdown/Measure    87
Data_type             5
Value               473
Value_Denominator   123
Value_Numerator     626
dtype: int64
```

```
# Summary of the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
 2   Title              974 non-null    object
 3   Period             974 non-null    object
 4   Country            974 non-null    object
 5   Age_group          974 non-null    object
 6   Breakdown/Measure  919 non-null    object
 7   Data_type          974 non-null    object
 8   Value              974 non-null    float64
 9   Value_Denominator  781 non-null    float64
 10  Value_Numerator    781 non-null    float64
dtypes: float64(3), object(8)
memory usage: 83.8+ KB
```

```
[ ] # Summary statistics of numerical columns
    df.describe()
```

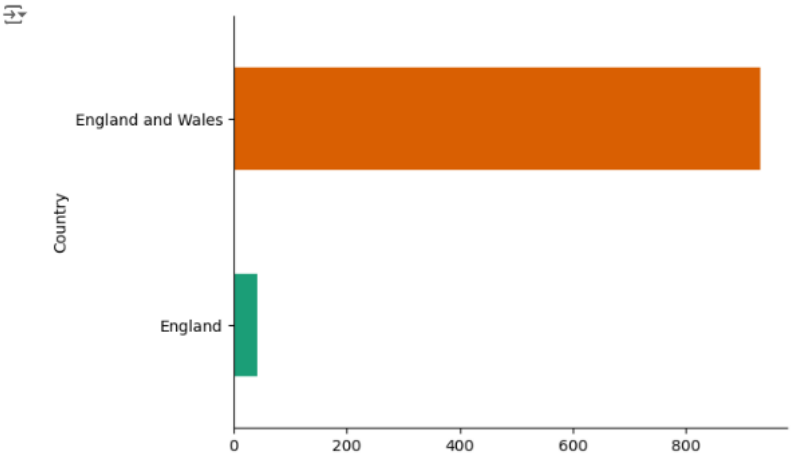
	Value	Value_Denominator	Value_Numerator
count	9.740000e+02	7.810000e+02	7.810000e+02
mean	5.694043e+04	8.419714e+05	3.170195e+05
std	3.619963e+05	2.009113e+06	1.122163e+06
min	0.000000e+00	4.500000e+01	0.000000e+00
25%	4.225000e+00	8.725000e+03	1.620000e+03
50%	2.190000e+01	1.187250e+05	2.031000e+04
75%	8.060000e+01	1.037340e+06	2.451750e+05
max	2.908670e+06	1.569696e+07	1.569696e+07

After completing the exploratory data analysis (EDA), we will now visualize the data to examine key aspects, including the distribution by country, period, and data type, as well as relationships such as country versus data type and period versus country. Additionally, we will review Table 1, which details the number of people with type 2 diabetes by age group and audit year.

Visualzations on the Dataset

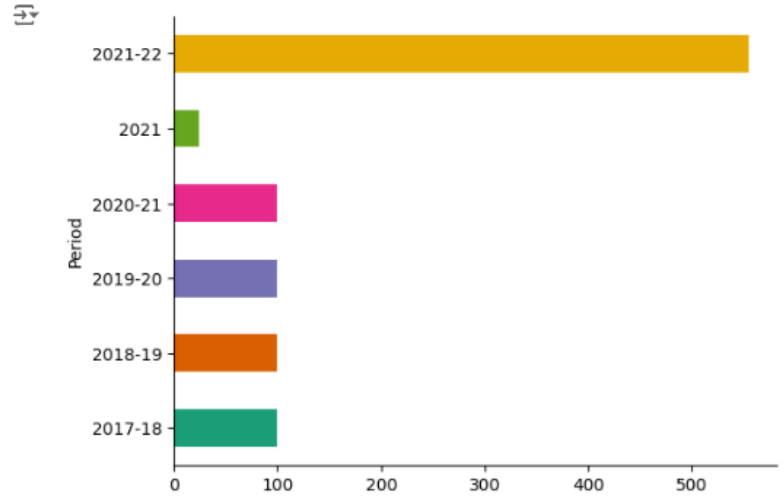
Country

```
[ ] # @title Country

    from matplotlib import pyplot as plt
    import seaborn as sns
    df.groupby('Country').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
    plt.gca().spines[['top', 'right',]].set_visible(False)
```



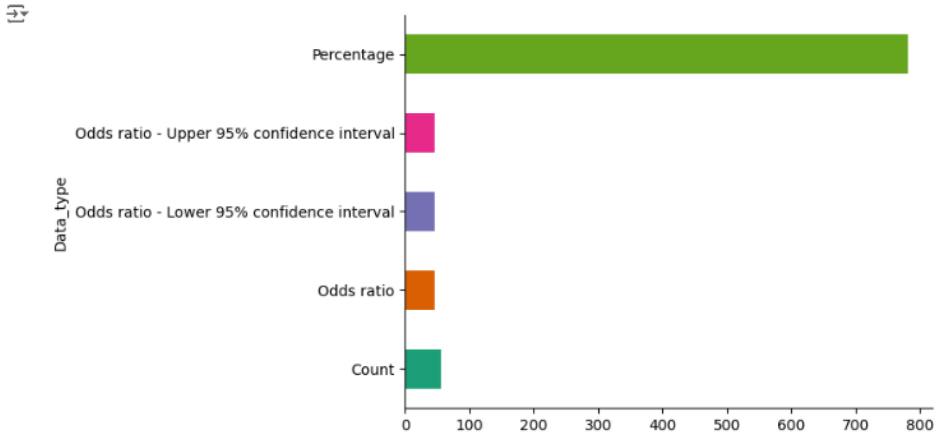
Period

```
[ ] # @title Period

    from matplotlib import pyplot as plt
    import seaborn as sns
    df.groupby('Period').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
    plt.gca().spines[['top', 'right',]].set_visible(False)
```



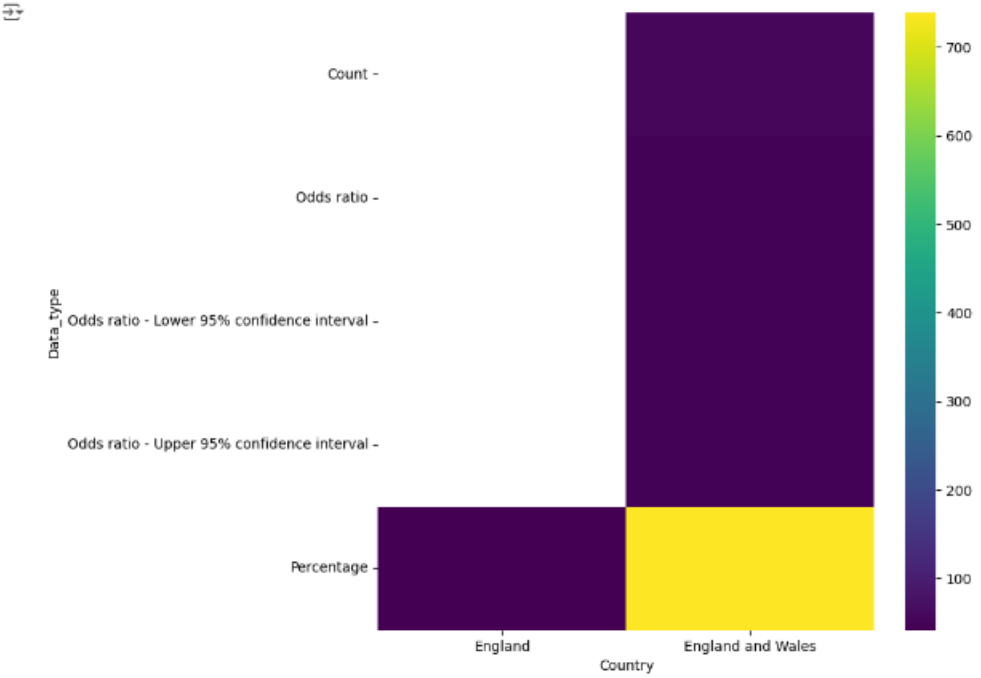
Data_type

```
[ ] # @title Data_type

    from matplotlib import pyplot as plt
    import seaborn as sns
    df.groupby('Data_type').size().plot(kind='barh', color=sns.palettes.mpl_palette('Dark2'))
    plt.gca().spines[['top', 'right',]].set_visible(False)
```



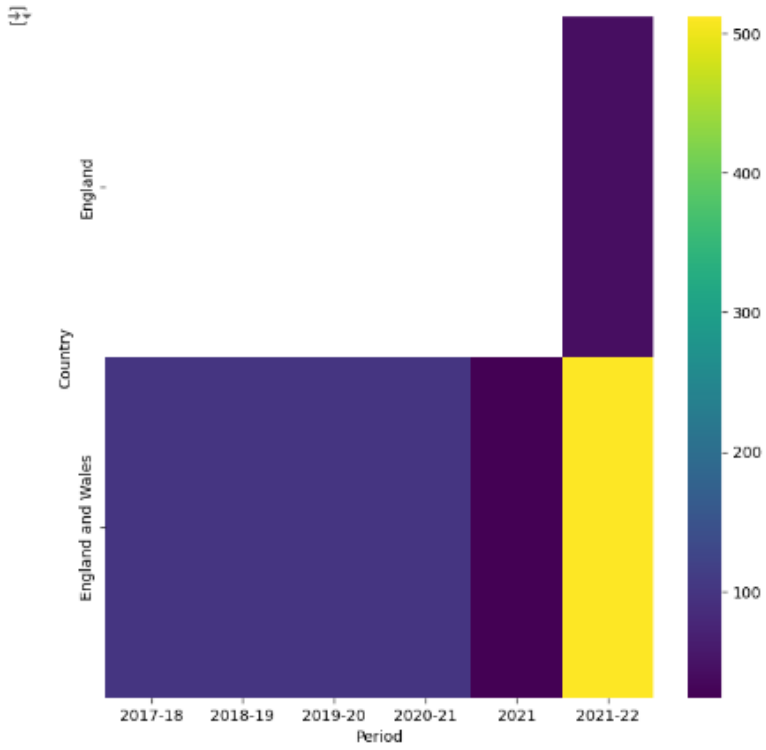
Country vs Data_type

```
[ ] # @title Country vs Data_type

    from matplotlib import pyplot as plt
    import seaborn as sns
    import pandas as pd
    plt.subplots(figsize=(8, 8))
    df_2dhist = pd.DataFrame({
        x_label: grp['Data_type'].value_counts()
        for x_label, grp in df.groupby('Country')
    })
    sns.heatmap(df_2dhist, cmap='viridis')
    plt.xlabel('Country')
    _ = plt.ylabel('Data_type')
```



Period vs Country

```
[ ]  # @title Period vs Country

     from matplotlib import pyplot as plt
     import seaborn as sns
     import pandas as pd
     plt.subplots(figsize=(8, 8))
     df_2dhist = pd.DataFrame({
         x_label: grp['Country'].value_counts()
         for x_label, grp in df.groupby('Period')
     })
     sns.heatmap(df_2dhist, cmap='viridis')
     plt.xlabel('Period')
     _ = plt.ylabel('Country')
```



Data : Table 1 - Number of people with type 2 diabetes by age group and audit year

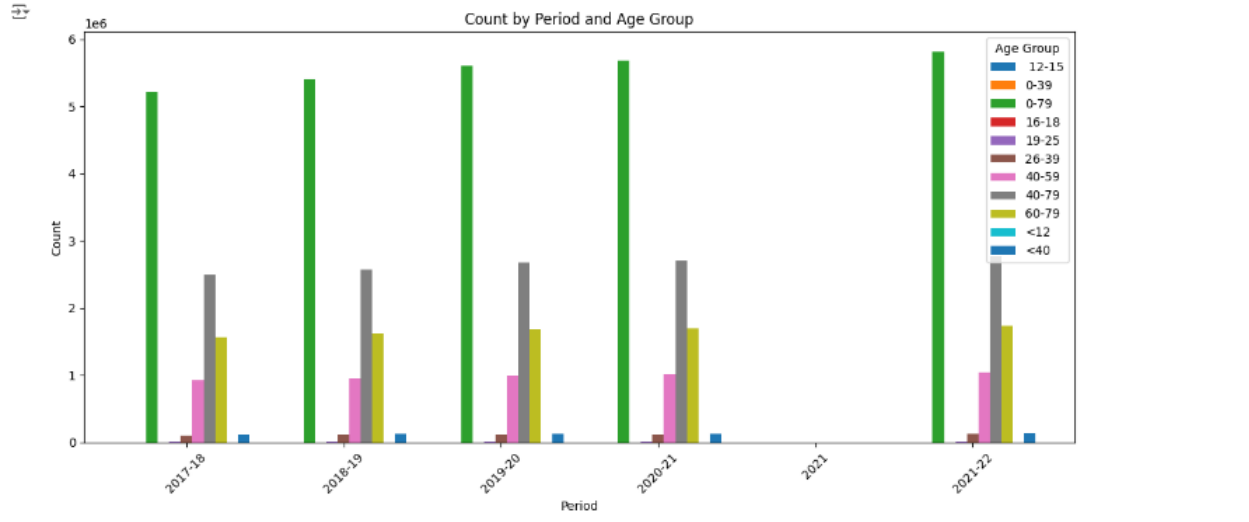
```
# Data : Table 1 - Number of people with type 2 diabetes by age group and audit year
d1 = df.iloc[:14,3:9]
d1
```

	Period	Country	Age_group	Breakdown/Measure	Data_type	Value
0	2017-18	England and Wales	<40	NaN	Count	117270.0
1	2018-19	England and Wales	<40	NaN	Count	123830.0
2	2019-20	England and Wales	<40	NaN	Count	129200.0
3	2020-21	England and Wales	<40	NaN	Count	132000.0
4	2021-22	England and Wales	<40	NaN	Count	139255.0
5	2017-18	England and Wales	40-79	NaN	Count	2488330.0
6	2018-19	England and Wales	40-79	NaN	Count	2577295.0
7	2019-20	England and Wales	40-79	NaN	Count	2674395.0
8	2020-21	England and Wales	40-79	NaN	Count	2705585.0
9	2021-22	England and Wales	40-79	NaN	Count	2769415.0
10	2017-18	England and Wales	0-79	NaN	Count	2605600.0
11	2018-19	England and Wales	0-79	NaN	Count	2701130.0
12	2019-20	England and Wales	0-79	NaN	Count	2803595.0
13	2020-21	England and Wales	0-79	NaN	Count	2837585.0

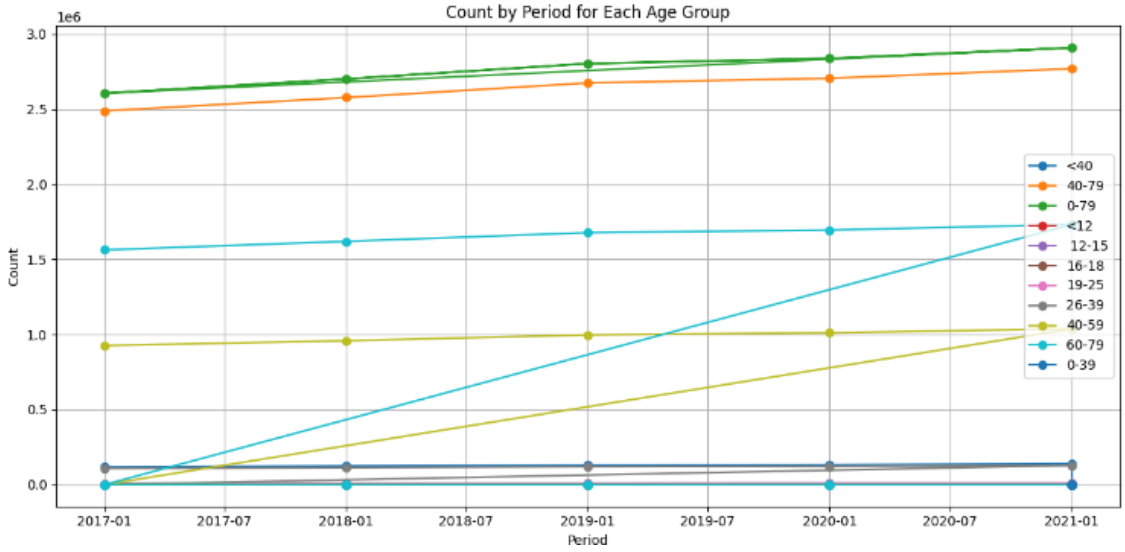
Visualizations on Table 1 - Number of people with type 2 diabetes by age group and audit year

Loading the dataset and grouping it by 'Period' and 'Age_group', and calculates the total count of cases for each group. It then creates a bar chart to visualize the number of people with type 2 diabetes by period and age group, with clear labels, a legend, and a title.

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the data
data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')

# Group by Period and Age_group, and calculate the sum of Value for each group
grouped_data = data.groupby(['Period', 'Age_group'])['Value'].sum().unstack()

# Plot the data
ax = grouped_data.plot(kind='bar', figsize=(12, 6), width=0.8)
plt.title('Count by Period and Age Group')
plt.xlabel('Period')
plt.ylabel('Count')
plt.legend(title='Age Group')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')

# Convert Period to datetime (use the first year of the period)
data['Period'] = pd.to_datetime(data['Period'].str[:4], format='%Y')

# 1. Line plot for all age groups
plt.figure(figsize=(12, 6))
for age_group in data['Age_group'].unique():
    age_data = data[data['Age_group'] == age_group]
    plt.plot(age_data['Period'], age_data['Value'], marker='o', label=age_group)

plt.title('Count by Period for Each Age Group')
plt.xlabel('Period')
plt.ylabel('Count')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# 2. Stacked bar plot
data_pivot = data.pivot_table(index='Period', columns='Age_group', values='Value', aggfunc='sum')
data_pivot.plot(kind='bar', stacked=True, figsize=(12, 6))
plt.title('Stacked Bar Plot of Counts by Age Group')
plt.xlabel('Period')
plt.ylabel('Count')
plt.legend(title='Age Group', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()


# 3. Box plot of values by age group
plt.figure(figsize=(10, 6))
sns.boxplot(x='Age_group', y='Value', data=data)
plt.title('Distribution of Values by Age Group')
plt.xlabel('Age Group')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```



Data Preprocessing and Model Training

Logistic Regression

Code handles categorical variables by encoding them into numerical values. It uses `LabelEncoder` from the `sklearn.preprocessing` module to transform each categorical column in the DataFrame `df` into numeric form, storing the encoders for potential inverse transformations later.

```
[ ] # Preprocessing: Handling categorical variables and missing values
    # Encoding categorical columns
    label_encoders = {}
    for column in df.select_dtypes(include=['object']).columns:
        le = LabelEncoder()
        df[column] = le.fit_transform(df[column].astype(str))
        label_encoders[column] = le
```

Using 'SimpleImputer', this code imputes missing values using the mean of each column. Using `StandardScaler`, it then scales all features to have a mean of 0 and a standard deviation of 1. The 'Diabetes_Status' column is also checked to see whether it exists; if not, a synthetic column representing the presence or absence of diabetes is created with random binary values, guaranteeing that the dataset includes a target variable for model training.

```
[ ] # Handling missing values
    imputer = SimpleImputer(strategy='mean')
    df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

```
[ ] # Feature scaling
    scaler = StandardScaler()
    df[df.columns] = scaler.fit_transform(df)
```

```
[ ] # Check if 'Diabetes_Status' column exists; if not, create it synthetically for this example
    if 'Diabetes_Status' not in df.columns:
        # For the purpose of this example, let's create a synthetic 'Diabetes_Status' column
        # Here, we randomly assign 0 or 1 to simulate the presence/absence of diabetes
        import numpy as np
        np.random.seed(42)
        df['Diabetes_Status'] = np.random.randint(0, 2, df.shape[0])
```

In order to prepare the dataset for training, this code defines the variables `X` (features) and `y` (target variable, 'Diabetes_Status'). The data is then divided into training and testing sets, with 20% set aside for testing. Ultimately, a logistic regression model is trained using the training data, and iterations up to 1000 are performed to ensure the model fits.

```
[ ] # Define X and y
    X = df.drop('Diabetes_Status', axis=1)
    y = df['Diabetes_Status']
```

```
[ ] # Split the dataset
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[ ] # Logistic Regression Model
    model = LogisticRegression(max_iter=1000)
    model.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression(max_iter=1000)
```

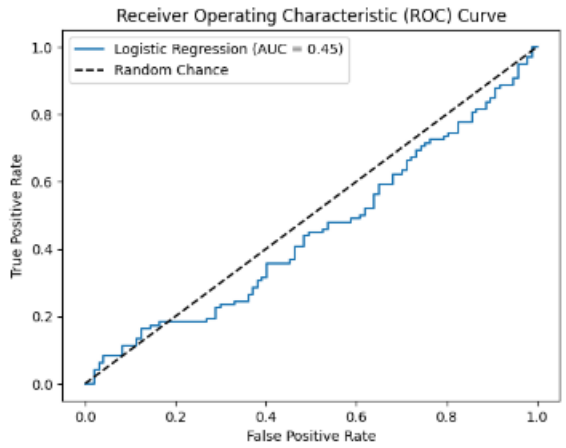
```
# Model Evaluation
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.4461538461538462
Classification Report:
              precision    recall  f1-score   support

           0       0.44      0.40      0.42        97
           1       0.45      0.49      0.47        98

    accuracy                           0.45       195
   macro avg       0.45      0.45      0.44       195
weighted avg       0.45      0.45      0.45       195
```

```
[ ] # Calculate AUC-ROC
    y_pred_proba = model.predict_proba(X_test)[:, 1]
    auc_roc = roc_auc_score(y_test, y_pred_proba)
    print("AUC-ROC:", auc_roc)
```

```
AUC-ROC: 0.44940037870818433
```

```
[ ] # Plot ROC Curve
    fpr, tpr, _ = roc_curve(y_test, y_pred_proba)
    plt.figure()
    plt.plot(fpr, tpr, label=f'Logistic Regression (AUC = {auc_roc:.2f})')
    plt.plot([0, 1], [0, 1], 'k--', label='Random Chance')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend()
    plt.show()
```



Decision Tree

Importing essential libraries for data manipulation, visualization, and machine learning. Then loading the dataset from a CSV file into a DataFrame and displays the first few rows to give an overview of the data's structure and contents.

```
[ ] import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.metrics import accuracy_score, classification_report, roc_auc_score, roc_curve
    from sklearn.preprocessing import LabelEncoder, StandardScaler, LabelBinarizer
    from sklearn.impute import SimpleImputer
```

```
[ ] # Load the dataset
    file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
    df = pd.read_csv(file_path)
```

```
[ ] # Display first few rows
    df.head()
```

	Section	Table/Figure	Title	Period	Country	Age_group	Breakdown/Measure	Data_type	Value	Value_Denominator	Value_Numerator
0	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2017-18	England and Wales	<40	NaN	Count	117270.0	NaN	NaN
1	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2018-19	England and Wales	<40	NaN	Count	123830.0	NaN	NaN
2	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2019-20	England and Wales	<40	NaN	Count	129200.0	NaN	NaN
3	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2020-21	England and Wales	<40	NaN	Count	132000.0	NaN	NaN
4	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2021-22	England and Wales	<40	NaN	Count	139255.0	NaN	NaN

```
[ ] # Summary of the DataFrame
    df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
 2   Title              974 non-null    object
 3   Period             974 non-null    object
 4   Country            974 non-null    object
 5   Age_group          974 non-null    object
 6   Breakdown/Measure  919 non-null    object
 7   Data_type          974 non-null    object
 8   Value              974 non-null    float64
 9   Value_Denominator  781 non-null    float64
 10  Value_Numerator    781 non-null    float64
dtypes: float64(3), object(8)
memory usage: 83.8+ KB
```

```
[ ] # Summary statistics of numerical columns
    df.describe()
```

	Value	Value_Denominator	Value_Numerator
count	9.740000e+02	7.810000e+02	7.810000e+02
mean	5.694043e+04	8.419714e+05	3.170195e+05
std	3.619963e+05	2.009113e+06	1.122163e+06
min	0.000000e+00	4.500000e+01	0.000000e+00
25%	4.225000e+00	8.725000e+03	1.620000e+03
50%	2.190000e+01	1.187250e+05	2.031000e+04
75%	8.060000e+01	1.037340e+06	2.451750e+05
max	2.908670e+06	1.569696e+07	1.569696e+07

```
[ ] # Preprocessing: Handling categorical variables and missing values
    # Encoding categorical columns
    label_encoders = {}
    for column in df.select_dtypes(include=['object']).columns:
        le = LabelEncoder()
        df[column] = le.fit_transform(df[column].astype(str))
        label_encoders[column] = le
```

```
[ ] # Handling missing values
    imputer = SimpleImputer(strategy='mean')
    df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

```
[ ] # Feature scaling
    scaler = StandardScaler()
    df[df.columns] = scaler.fit_transform(df)
```

In order to train a decision tree model, this code defines the variables `X` (features) and `y` (target variable, 'Age_group'). Training a decision tree classifier on the training data involves encoding the target variable as categorical data, dividing the dataset into training and testing sets, and setting aside 20% for testing.

```
[ ] # Define X and y
    X = df.drop('Age_group', axis=1)
    y = df['Age_group']
```

```
[ ] # Ensure target variable is treated as categorical
    le_y = LabelEncoder()
    y = le_y.fit_transform(y)
```

```
[ ] # Split the dataset
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[ ] # Decision Tree Model
    model = DecisionTreeClassifier()
    model.fit(X_train, y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
# Model Evaluation
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("Classification Report:")
print(classification_report(y_test, y_pred))
```

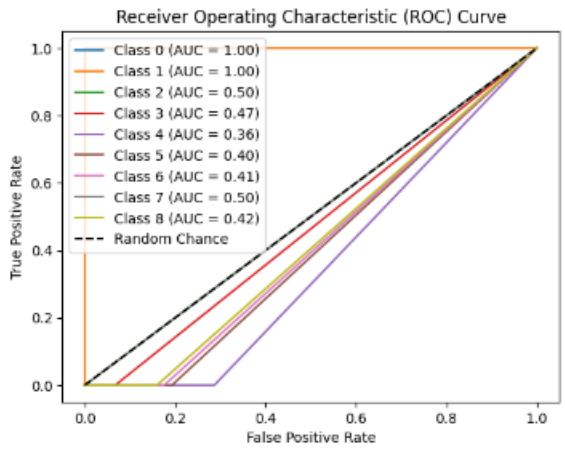
```
Accuracy: 0.9948717948717949
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         6
           1       1.00      1.00      1.00        30
           3       1.00      1.00      1.00        10
           4       1.00      1.00      1.00        47
           5       0.97      1.00      0.98        31
           6       1.00      1.00      1.00        29
           8       1.00      1.00      1.00        30
           9       1.00      1.00      1.00         5
          10       1.00      0.86      0.92         7

    accuracy                           0.99       195
   macro avg       1.00      0.98      0.99       195
weighted avg       1.00      0.99      0.99       195
```

```
# Calculate AUC-ROC for each class using LabelBinarizer
lb = LabelBinarizer()
y_test_binarized = lb.fit_transform(y_test)
y_pred_proba = model.predict_proba(X_test)
```

```
# Check the number of classes
print(f"Number of classes in y_test: {len(lb.classes_)}")
print(f"Shape of y_pred_proba: {y_pred_proba.shape}")
```

```
Number of classes in y_test: 9
Shape of y_pred_proba: (195, 11)
```

```
# Plot ROC Curve for each class
fpr = {}
tpr = {}
plt.figure()
for i in range(len(lb.classes_)):
    fpr[i], tpr[i], _ = roc_curve(y_test_binarized[:, i], y_pred_proba[:, i])
    plt.plot(fpr[i], tpr[i], label=f'Class {i} (AUC = {roc_auc_score(y_test_binarized[:, i], y_pred_proba[:, i]):.2f})')

plt.plot([0, 1], [0, 1], 'k--', label='Random Chance')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()
```



Support Vector Machine

Importing essential libraries for data manipulation, visualization, and machine learning. Then loading the dataset from a CSV file into a DataFrame and displays the first few rows to give an overview of the data's structure and contents.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, roc_auc_score, roc_curve
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
```

```
# Load the dataset
file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
df = pd.read_csv(file_path)
```

```
# Display first few rows
df.head()
```

	Section	Table/Figure	Title	Period	Country	Age_group	Breakdown/Measure	Data_type	Value	Value_Denominator	Value_Numerator
0	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2017-18	England and Wales	<40	NaN	Count	117270.0	NaN	NaN
1	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2018-19	England and Wales	<40	NaN	Count	123830.0	NaN	NaN
2	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2019-20	England and Wales	<40	NaN	Count	129200.0	NaN	NaN
3	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2020-21	England and Wales	<40	NaN	Count	132000.0	NaN	NaN
4	Characteristics and trends	Table 1	Number of people with type 2 diabetes by age g...	2021-22	England and Wales	<40	NaN	Count	139255.0	NaN	NaN

```
# Summary of the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
 2   Title              974 non-null    object
 3   Period             974 non-null    object
 4   Country            974 non-null    object
 5   Age_group          974 non-null    object
 6   Breakdown/Measure  919 non-null    object
 7   Data_type          974 non-null    object
 8   Value              974 non-null    float64
 9   Value_Denominator  781 non-null    float64
 10  Value_Numerator    781 non-null    float64
dtypes: float64(3), object(8)
memory usage: 83.8+ KB
```

```
# Summary statistics of numerical columns
df.describe()
```

	Value	Value_Denominator	Value_Numerator
count	9.740000e+02	7.810000e+02	7.810000e+02
mean	5.694043e+04	8.419714e+05	3.170195e+05
std	3.619963e+05	2.009113e+06	1.122163e+06
min	0.000000e+00	4.500000e+01	0.000000e+00
25%	4.225000e+00	8.725000e+03	1.620000e+03
50%	2.190000e+01	1.187250e+05	2.031000e+04
75%	8.060000e+01	1.037340e+06	2.451750e+05
max	2.908670e+06	1.569696e+07	1.569696e+07

```
# Preprocessing: Handling categorical variables and missing values
# Encoding categorical columns
label_encoders = {}
for column in df.select_dtypes(include=['object']).columns:
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column].astype(str))
    label_encoders[column] = le
```

```
# Handling missing values
imputer = SimpleImputer(strategy='mean')
df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

```
# Feature scaling
scaler = StandardScaler()
df[df.columns] = scaler.fit_transform(df)
```

```
[ ] # Define X and y
    X = df.drop('Age_group', axis=1)
    y = df['Value']
```

```
# Ensure target variable is treated as categorical
y = LabelEncoder().fit_transform(y)
```

```
[ ] # Split the dataset
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[ ] # Support Vector Machine Model
    model = SVC(probability=True)
    model.fit(X_train, y_train)
```

```
SVC
SVC(probability=True)
```

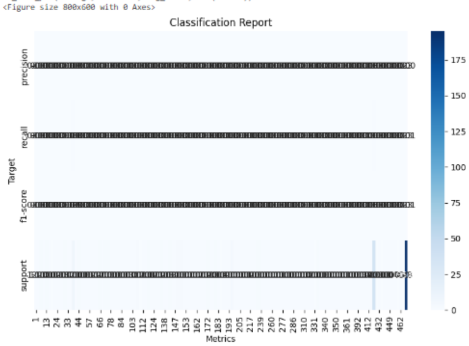
Code uses the test set's target variable as a predictor to assess how well the trained Support Vector Machine (SVM) model performs. A classification report containing the precision, recall, and F1-score for each class is printed along with the accuracy, which is estimated to be 22.6%. To address situations when a class may not be predicted at all, `zero_division=1} is used.

```
[ ] # Model Evaluation
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print("Accuracy:", accuracy)
    print("Classification Report:")
    print(classification_report(y_test, y_pred, zero_division=1))
```

```
Accuracy: 0.22564102564102564
Classification Report:
              precision    recall  f1-score   support

           1       1.00      0.00      0.00         1
           2       1.00      0.00      0.00         1
```

```
# Plot ROC Curve for each class using One-vs-Rest (OvR) strategy
# Plotting the classification report
def plot_classification_report(y_true, y_pred, ax=None):
    report = classification_report(y_true, y_pred, output_dict=True)
    df_report = pd.DataFrame(report).transpose()
    if ax is None:
        fig, ax = plt.subplots(figsize=(10, 6))
    sns.heatmap(df_report.iloc[:-1, :].T, annot=True, ax=ax, cmap='Blues', fmt=".2f")
    ax.set_title('Classification Report')
    ax.set_xlabel('Metrics')
    ax.set_ylabel('Target')
    return ax

plt.figure(figsize=(8, 6))
plot_classification_report(y_test, y_pred)
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471: UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0 in labels with no true samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
<Figure size 800x600 with 0 Axes>
```



Neural Networks

Importing essential libraries for data manipulation, visualization, and machine learning. Then loading the dataset from a CSV file into a DataFrame and displays the first few rows to give an overview of the data's structure and contents.

```
[ ] import pandas as pd
    from sklearn.model_selection import train_test_split
    from sklearn.neural_network import MLPClassifier
    from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
    from sklearn.preprocessing import LabelEncoder, StandardScaler
    from sklearn.impute import SimpleImputer
    from sklearn.exceptions import UndefinedMetricWarning
    import matplotlib.pyplot as plt
    import seaborn as sns
    import warnings
```

```
[ ] # Load the dataset
    file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
    df = pd.read_csv(file_path)
```

```
[ ] # Display first few rows
    print(df.head())
```

```
                    Section Table/Figure  \
0  Characteristics and trends      Table 1
1  Characteristics and trends      Table 1
2  Characteristics and trends      Table 1
3  Characteristics and trends      Table 1
```

```
[ ] # Summary of the DataFrame
    print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
 2   Title              974 non-null    object
 3   Period             974 non-null    object
 4   Country            974 non-null    object
 5   Age_group          974 non-null    object
 6   Breakdown/Measure  919 non-null    object
 7   Data_type          974 non-null    object
 8   Value              974 non-null    float64
 9   Value_Denominator  781 non-null    float64
 10  Value_Numerator    781 non-null    float64
dtypes: float64(3), object(8)
memory usage: 83.8+ KB
None
```

```
[ ] # Summary statistics of numerical columns
    print(df.describe())
```

```
              Value  Value_Denominator  Value_Numerator
count  9.740000e+02       7.810000e+02     7.810000e+02
mean   5.694043e+04       8.419714e+05     3.170195e+05
std    3.619963e+05       2.009113e+06     1.122163e+06
min    0.000000e+00       4.500000e+01     0.000000e+00
25%    4.225000e+00       8.725000e+03     1.620000e+03
50%    2.190000e+01       1.187250e+05     2.031000e+04
75%    8.060000e+01       1.037340e+06     2.451750e+05
max    2.908670e+06       1.569696e+07     1.569696e+07
```

```
[ ] # Encoding categorical columns
    label_encoders = {}
    for column in df.select_dtypes(include=['object']).columns:
        le = LabelEncoder()
        df[column] = le.fit_transform(df[column].astype(str))
        label_encoders[column] = le
```

```
[ ] # Handling missing values
    imputer = SimpleImputer(strategy='mean')
    df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

```
[ ] # Feature scaling
    scaler = StandardScaler()
    df[df.columns] = scaler.fit_transform(df)
```

```
[ ] # Assuming 'Title' is the target variable
    X = df.drop('Title', axis=1)
    y = df['Title']
```

```
[ ] # Convert to string type to ensure it's treated as categorical
    y = df['Title'].astype(str)
```

```
[ ] # Splitting the data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[ ] # Neural Network Model
    model = MLPClassifier(hidden_layer_sizes=(100,), max_iter=1000, random_state=42)
```

```
[ ] # Fit the model
    model.fit(X_train, y_train)
```

```
▾ MLPClassifier
MLPClassifier(max_iter=1000, random_state=42)
```

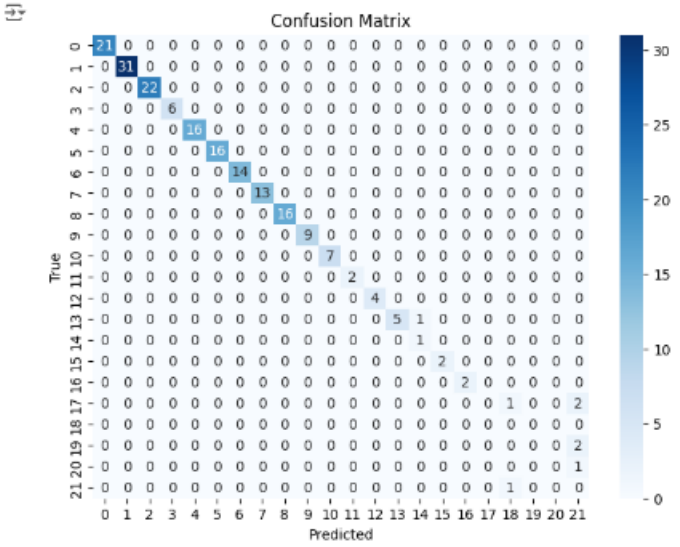
```
[ ] # Suppress UndefinedMetricWarning
    warnings.filterwarnings("ignore", category=UndefinedMetricWarning)
```

Through prediction of the test set's target variable, this function assesses how well the trained neural network model performs. It computes and outputs the accuracy, which is around 95.9%, and offers a thorough classification report with precision, recall, and F1-score for every class. To handle divisions by zero in the metrics, use {zero_division=1}.

```
[ ] # Model Evaluation
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print("Accuracy:", accuracy)
    print("Classification Report:")
    print(classification_report(y_test, y_pred, zero_division=1))
```

```
Accuracy: 0.958974358974359
Classification Report:
                     precision    recall  f1-score   support

-0.08391213104986046       1.00      1.00      1.00        21
-0.29187756525485836       1.00      1.00      1.00        31
 -0.4998429994598563       1.00      1.00      1.00        22
 -0.7078084336648541       1.00      1.00      1.00         6
```

```
[ ] # Plotting Confusion Matrix
    plt.figure(figsize=(8, 6))
    plt.title('Confusion Matrix')
    sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('True')
    plt.show()
```



Ensemble Methods

Importing essential libraries for data manipulation, visualization, and machine learning. Then loading the dataset from a CSV file into a DataFrame and displays the first few rows to give an overview of the data's structure and contents.

```
[ ] import pandas as pd
    import seaborn as sns
    import matplotlib.pyplot as plt
    from sklearn.model_selection import train_test_split
    from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, VotingClassifier
    from sklearn.metrics import accuracy_score, classification_report
    from sklearn.preprocessing import LabelEncoder, StandardScaler
    from sklearn.impute import SimpleImputer
```

```
[ ] # Load the dataset
    file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
    df = pd.read_csv(file_path)
```

```
    # Display first few rows
    print(df.head())
```

```
                    Section Table/Figure  \
0  Characteristics and trends      Table 1
1  Characteristics and trends      Table 1
2  Characteristics and trends      Table 1
```

```
    # Summary of the DataFrame
    print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
 2   Title              974 non-null    object
 3   Period             974 non-null    object
 4   Country            974 non-null    object
 5   Age_group          974 non-null    object
 6   Breakdown/Measure  919 non-null    object
 7   Data_type          974 non-null    object
 8   Value              974 non-null    float64
 9   Value_Denominator  781 non-null    float64
 10  Value_Numerator    781 non-null    float64
dtypes: float64(3), object(8)
memory usage: 83.8+ KB
None
```

```
[ ] # Summary statistics of numerical columns
    print(df.describe())
```

```
              Value  Value_Denominator  Value_Numerator
count  9.740000e+02       7.810000e+02     7.810000e+02
mean   5.694043e+04       8.419714e+05     3.170195e+05
std    3.619963e+05       2.009113e+06     1.122163e+06
min    0.000000e+00       4.500000e+01     0.000000e+00
25%    4.225000e+00       8.725000e+03     1.620000e+03
50%    2.190000e+01       1.187250e+05     2.031000e+04
75%    8.060000e+01       1.037340e+06     2.451750e+05
max    2.908670e+06       1.569696e+07     1.569696e+07
```

```
[ ] # Preprocessing: Handling categorical variables and missing values
    # Encoding categorical columns
    label_encoders = {}
    for column in df.select_dtypes(include=['object']).columns:
        le = LabelEncoder()
        df[column] = le.fit_transform(df[column].astype(str))
        label_encoders[column] = le
```

```
[ ] # Handling missing values
    imputer = SimpleImputer(strategy='mean')
    df = pd.DataFrame(imputer.fit_transform(df), columns=df.columns)
```

```
[ ] # Feature scaling
    scaler = StandardScaler()
    df[df.columns] = scaler.fit_transform(df)
```

```
[ ] # Assuming 'Title' is the target variable
    X = df.drop('Title', axis=1)
    y = df['Title']
```

```
[ ] # Convert target variable to discrete classes
    # Example: If 'Title' is continuous, convert it to categorical
    # For demonstration, let's convert it to integer labels
    y = y.astype(int)
```

```
[ ] # Split data into train and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
[ ] # Ensemble Methods: RandomForest, GradientBoosting, VotingClassifier
    rf_model = RandomForestClassifier(random_state=42)
    gb_model = GradientBoostingClassifier(random_state=42)
```

```
[ ] # Voting Classifier
    voting_model = VotingClassifier(estimators=[('rf', rf_model), ('gb', gb_model)], voting='soft')
```

```
    # Fit the VotingClassifier
    voting_model.fit(X_train, y_train)
```

```
VotingClassifier
    rf
        RandomForestClassifier
    gb
        GradientBoostingClassifier
```

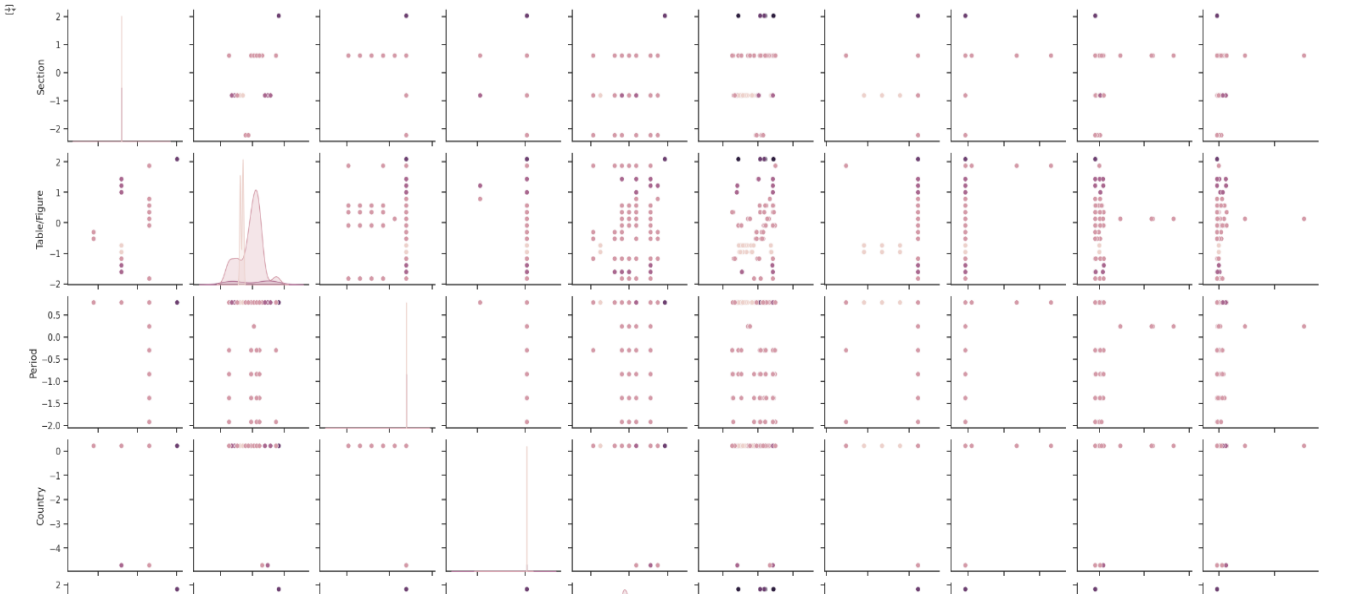
This function predicts the target variable on the test set in order to assess the `VotingClassifier`s` performance. It presents a comprehensive classification report with a 100% accuracy rate, demonstrating that the model accurately identifies every test sample.

```
[ ] # Model Evaluation
    y_pred = voting_model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print("Accuracy:", accuracy)
    print("Classification Report:")
    print(classification_report(y_test, y_pred))
```

```
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

          -1       1.00      1.00      1.00        30
           0       1.00      1.00      1.00       143
           1       1.00      1.00      1.00        15
           2       1.00      1.00      1.00         3
           3       1.00      1.00      1.00         4

    accuracy                           1.00       195
   macro avg       1.00      1.00      1.00       195
weighted avg       1.00      1.00      1.00       195
```

```
[ ] # Combine predicted labels and features for visualization
    result_df = pd.DataFrame(X_test, columns=X.columns)
    result_df['Predicted_Title'] = y_pred
```

```
# Visualize using pairplot
sns.set(style="ticks")  # Set the seaborn style to "ticks"
sns.pairplot(result_df, hue='Predicted_Title', diag_kind='kde')
plt.show()
```



Support Vector Machine Regressor

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import cross_val_score, train_test_split
from sklearn.preprocessing import OneHotEncoder, LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.svm import SVR
from sklearn.metrics import mean_absolute_error,r2_score

# Load the dataset
file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
data = pd.read_csv(file_path)

# Display basic information about the dataset
print(data.info())
print("\nFirst few rows of the data:")
print(data.head())

# Extract relevant portion of the dataset (rows 57 to 116)
data_subset = data.iloc[57:117]

# Display the first few rows of the subset
print(data_subset.head())

# Selecting features and target variable
features =  ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
target = 'Value'

# Splitting the data into features and target
X = data_subset[features]
y = data_subset[target]

# Handling missing values and categorical encoding
categorical_features = features
numerical_features = []

# Preprocessing pipeline
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features)
    ])

# Define the model
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('imputer', SimpleImputer(strategy='mean')),
    ('regressor', SVR(kernel='linear'))
])

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train the model
model.fit(X_train, y_train)

# Predict and evaluate
y_pred = model.predict(X_test)
SVM_mae = mean_absolute_error(y_test, y_pred)
SVM_r2 = r2_score(y_test, y_pred)
print("Mean Absolute Error:", SVM_mae)
print("Mean Absolute Error:", SVM_r2)

# Convert the Period column to string
data_subset['Period'] = data_subset['Period'].astype(str)
```

```
# Calculate year-over-year growth
data_subset['Year'] = data_subset['Period'].str[:4].astype(int)
data_subset['Growth'] = data_subset['Value'].pct_change() * 100

# Calculate total growth from 2017-18 to 2021-22
total_growth = (data_subset['Value'].iloc[-1] - data_subset['Value'].iloc[0]) / data_subset['Value'].iloc[0] * 100

# Create a line plot to show the trend of type 2 diabetes cases over time
plt.figure(figsize=(12, 6))
sns.lineplot(x='Period', y='Value', data=data_subset, marker='o', linewidth=2, markersize=10)
plt.title('Trend of Type 2 Diabetes Cases (Age < 40) in England and Wales', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Number of People with Type 2 Diabetes', fontsize=12)
plt.xticks(rotation=45)
plt.grid(True, linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# Create a summary statistics visualization
plt.figure(figsize=(12, 6))
plt.text(0.5, 0.8, f"Total Cases in 2021-22: {data_subset['Value'].iloc[-1]:,.0f}",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.text(0.5, 0.5, f"Total Growth from 2017-18 to 2021-22: {total_growth:.2f}%",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.text(0.5, 0.2, f"Average Annual Growth Rate: {data_subset['Growth'].mean():.2f}%",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.axis('off')
plt.title('Summary Statistics for Type 2 Diabetes Cases (Age < 40)', fontsize=18)
plt.tight_layout()
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 974 entries, 0 to 973
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Section            974 non-null    object
 1   Table/Figure       935 non-null    object
```

Multi-Layer Perceptron Regressor (Neural Network)

```
[ ] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.model_selection import train_test_split, cross_val_score
    from sklearn.preprocessing import OneHotEncoder
    from sklearn.impute import SimpleImputer
    from sklearn.compose import ColumnTransformer
    from sklearn.pipeline import Pipeline
    from sklearn.svm import SVR
    from sklearn.neural_network import MLPRegressor
    from sklearn.metrics import mean_absolute_error,r2_score

    # Load and prepare the data
    data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')
    data_subset = data.iloc[57:117]

    #   Select features and target variable
    features = ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
    target = 'Value'

    X = data_subset[features]
    y = data_subset[target]

    #   Visualize the distribution of the target variable
    plt.figure(figsize=(10, 6))
    sns.histplot(y, kde=True, color='skyblue')
    plt.title('Distribution of Target Variable (Value)')
    plt.xlabel('Value')
    plt.ylabel('Frequency')
    plt.show()

    #   Preprocess data and define the model pipeline
    categorical_features = features
    numerical_features = []

    preprocessor = ColumnTransformer(
        transformers=[
            ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features)
        ])

    model = Pipeline(steps=[
        ('preprocessor', preprocessor),
        ('imputer', SimpleImputer(strategy='mean')),
        ('regressor', MLPRegressor(hidden_layer_sizes=(100, 50), max_iter=1000, random_state=42))
    ])

    #   Perform cross-validation and print results
    cv_scores = cross_val_score(model, X, y, cv=5, scoring='neg_mean_absolute_error')
    cv_scores = -cv_scores
    print("Cross-Validation MAE Scores:", cv_scores)
    ML_regressor = cv_scores.mean()
    print("Mean CV MAE:", ML_regressor)
    ML_regressor_r2 = r2_score(y_test, y_pred)
    print("R2 score:", ML_regressor_r2)

    #   Visualize cross-validation results
    plt.figure(figsize=(10, 6))
    sns.boxplot(x=cv_scores, color='lightgreen')
    plt.title('Cross-Validation MAE Scores')
    plt.xlabel('Mean Absolute Error')
    plt.show()
```

```
#   Analyze feature importance (for categorical features)
encoder = OneHotEncoder(handle_unknown='ignore')
encoded_features = encoder.fit_transform(X)
feature_names = encoder.get_feature_names_out(features)

plt.figure(figsize=(12, 8))
sns.barplot(x=np.sum(encoded_features.toarray(), axis=0), y=feature_names, palette='viridis')
plt.title('Feature Importance (Based on Frequency)')
plt.xlabel('Frequency')
plt.ylabel('Features')
plt.show()

#   Visualize relationships between features and target
fig, axes = plt.subplots(2, 2, figsize=(20, 15))
fig.suptitle('Relationship between Selected Features and Target Variable', fontsize=16)

for i, feature in enumerate(['Country', 'Age_group', 'Period', 'Breakdown/Measure']):
    ax = axes[i // 2, i % 2]
    sns.boxplot(x=feature, y=target, data=data_subset, ax=ax, palette='Set3')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha='right')
    ax.set_title(f'{feature} vs {target}')

plt.tight_layout()
plt.show()
```

Decision Tree Regressor

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error,r2_score
from sklearn.inspection import permutation_importance

#   Load and prepare the data
data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')
data_subset = data.iloc[57:117]

features = ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
target = 'Value'

X = data_subset[features]
y = data_subset[target]

#   Create a correlation heatmap for categorical features
plt.figure(figsize=(12, 10))
sns.heatmap(pd.get_dummies(X).corr(), annot=False, cmap='coolwarm', center=0)
plt.title('Correlation Heatmap of Encoded Features')
plt.tight_layout()
plt.show()

#   Preprocess data and define the model pipeline
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), features)
    ])

model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('imputer', SimpleImputer(strategy='mean')),
    ('regressor', DecisionTreeRegressor(random_state=42))
])

#   Perform cross-validation and visualize results
cv_scores = cross_val_score(model, X, y, cv=5, scoring='neg_mean_absolute_error')
cv_scores = -cv_scores
print("Decision Tree Regressor Cross-Validation MAE Scores:", cv_scores)
DTR_mae = cv_scores.mean()
print("Mean CV MAE:", DTR_mae)
DTR_r2 = r2_score(y_test, y_pred)
print("R2 score:", DTR_r2)

plt.figure(figsize=(10, 6))
sns.violinplot(y=cv_scores, color='orange')
plt.title('Distribution of Cross-Validation MAE Scores')
plt.ylabel('Mean Absolute Error')
plt.show()

# Train the model and analyze feature importance
model.fit(X, y)
feature_importance = permutation_importance(model, X, y, n_repeats=10, random_state=42)

sorted_idx = feature_importance.importances_mean.argsort()
sorted_features = [features[i] for i in sorted_idx]
sorted_importance = feature_importance.importances_mean[sorted_idx]
```

```
plt.figure(figsize=(12, 8))
sns.barplot(x=sorted_importance, y=sorted_features, palette='YlOrRd')
plt.title('Feature Importance (Permutation Importance)')
plt.xlabel('Mean Importance')
plt.tight_layout()
plt.show()

#   Visualize the distribution of target variable by categorical features
fig, axes = plt.subplots(2, 2, figsize=(20, 15))
fig.suptitle('Distribution of Target Variable by Categorical Features', fontsize=16)

for i, feature in enumerate(['Country', 'Age_group', 'Period', 'Breakdown/Measure']):
    ax = axes[i // 2, i % 2]
    sns.violinplot(x=feature, y=target, data=data_subset, ax=ax, palette='Set2')
    ax.set_xticklabels(ax.get_xticklabels(), rotation=45, ha='right')
    ax.set_title(f'{feature} vs {target}')

plt.tight_layout()
plt.show()

#   Analyze prediction errors
y_pred = model.predict(X)
errors = y - y_pred

plt.figure(figsize=(12, 6))
sns.scatterplot(x=y, y=errors, hue=data_subset['Country'], palette='deep')
plt.title('Prediction Errors vs Actual Values')
plt.xlabel('Actual Values')
plt.ylabel('Prediction Errors')
plt.legend(title='Country', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

Random Forest Regressor

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, r2_score

# Load and prepare the data
data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')
data_subset = data.iloc[57:117]

features = ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
target = 'Value'

X = data_subset[features]
y = data_subset[target]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Visualize the distribution of the target variable with a density plot
plt.figure(figsize=(10, 6))
sns.kdeplot(data=y, shade=True, color='purple')
plt.title('Density Plot of Target Variable (Value)')
plt.xlabel('Value')
plt.ylabel('Density')
plt.show()

# Preprocess data and define the model pipeline
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), features)
    ])

model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('imputer', SimpleImputer(strategy='mean')),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])

# Perform cross-validation and visualize results
cv_scores = cross_val_score(model, X, y, cv=5, scoring='neg_mean_absolute_error')
cv_scores = -cv_scores
print("Random Forest Regressor Cross-Validation MAE Scores:", cv_scores)
RFR_mae = cv_scores.mean()
print("Mean CV MAE:", RFR_mae)

plt.figure(figsize=(10, 6))
sns.swarmplot(y=cv_scores, color='green', size=10)
plt.title('Swarm Plot of Cross-Validation MAE Scores')
plt.ylabel('Mean Absolute Error')
plt.show()

# Train the model and analyze feature importance
model.fit(X_train, y_train)
feature_importance = model.named_steps['regressor'].feature_importances_
feature_names = model.named_steps['preprocessor'].get_feature_names_out()

importance_df = pd.DataFrame({'feature': feature_names, 'importance': feature_importance})
importance_df = importance_df.sort_values('importance', ascending=False).head(10)
```

```
plt.figure(figsize=(12, 8))
sns.barplot(x='importance', y='feature', data=importance_df, palette='viridis')
plt.title('Top 10 Feature Importances (Random Forest)')
plt.xlabel('Importance')
plt.tight_layout()
plt.show()

# Analyze prediction errors with residual plot
y_pred = model.predict(X_test)
residuals = y_test - y_pred
Random_forest_mae = mean_absolute_error(y_test, y_pred)
y_range = y_test.max() - y_test.min()
normalized_accuracy = 1 - (Random_forest_mae / y_range)
print(f"Normalized accuracy: {normalized_accuracy:.4f}")

RFR_r2 = r2_score(y_test, y_pred)
print("R2 score:", RFR_r2)

plt.figure(figsize=(12, 6))
sns.residplot(x=y_pred, y=residuals, lowess=True, scatter_kws={'alpha': 0.5}, line_kws={'color': 'red', 'lw': 2})
plt.title('Residual Plot')
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.tight_layout()
plt.show()

# Visualize feature interactions
fig, ax = plt.subplots(figsize=(12, 8))
sns.scatterplot(x=X_test['Age_group'], y=y_test, hue=X_test['Country'], size=X_test['Period'], sizes=(20, 200), palette='coolwarm', ax=ax)
plt.title('Feature Interactions: Age Group, Country, and Period')
plt.xlabel('Age Group')
plt.ylabel('Value')
plt.legend(title='Country', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```

Logistic Regression Regressor

```
[ ] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
    from sklearn.model_selection import cross_val_score, train_test_split
    from sklearn.preprocessing import OneHotEncoder, StandardScaler
    from sklearn.impute import SimpleImputer
    from sklearn.compose import ColumnTransformer
    from sklearn.pipeline import Pipeline
    from sklearn.linear_model import LogisticRegression
    from sklearn.metrics import accuracy_score, classification_report

    # Load the dataset
    file_path = 'NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv'
    data = pd.read_csv(file_path)

    # Display basic information about the dataset
    print(data.info())
    print("\nFirst few rows of the data:")
    print(data.head())

    # Extract relevant portion of the dataset (rows 57 to 116)
    data_subset = data.iloc[57:117]

    # Display the first few rows of the subset
    print(data_subset.head())

    # Selecting features and target variable
    features = ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
    target = 'Value'

    # Splitting the data into features and target
    X = data_subset[features]
    y = data_subset[target]

    # Convert the target variable to binary for logistic regression
    y_binary = (y > y.median()).astype(int)

    # Identify numeric and categorical columns
    numeric_features = X.select_dtypes(include=['int64', 'float64']).columns
    categorical_features = X.select_dtypes(include=['object']).columns

    # Create preprocessor
    preprocessor = ColumnTransformer(
        transformers=[
            ('num', StandardScaler(), numeric_features),
            ('cat', Pipeline([
                ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
                ('onehot', OneHotEncoder(handle_unknown='ignore'))
            ]), categorical_features)
        ])

    # Define the model
    model = Pipeline([
        ('preprocessor', preprocessor),
        ('classifier', LogisticRegression(random_state=42))
    ])

    # Split the data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X, y_binary, test_size=0.2, random_state=42)

    # Train the model
    model.fit(X_train, y_train)
```

```
# Predict and evaluate
y_pred = model.predict(X_test)
Logistoc_regression_accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Convert the Period column to string
data_subset['Period'] = data_subset['Period'].astype(str)

# Calculate year-over-year growth
data_subset['Year'] = data_subset['Period'].str[:4].astype(int)
data_subset['Growth'] = data_subset['Value'].pct_change() * 100

# Calculate total growth from 2017-18 to 2021-22
total_growth = (data_subset['Value'].iloc[-1] - data_subset['Value'].iloc[0]) / data_subset['Value'].iloc[0] * 100

# Create a line plot to show the trend of type 2 diabetes cases over time
plt.figure(figsize=(12, 6))
sns.lineplot(x='Period', y='Value', data=data_subset, marker='o', linewidth=2, markersize=10)
plt.title('Trend of Type 2 Diabetes Cases (Age < 40) in England and Wales', fontsize=16)
plt.xlabel('Year', fontsize=12)
plt.ylabel('Number of People with Type 2 Diabetes', fontsize=12)
plt.xticks(rotation=45)
plt.grid(True, linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# Create a summary statistics visualization
plt.figure(figsize=(12, 6))
plt.text(0.5, 0.8, f"Total Cases in 2021-22: {data_subset['Value'].iloc[-1]:,.0f}",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.text(0.5, 0.5, f"Total Growth from 2017-18 to 2021-22: {total_growth:.2f}%",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.text(0.5, 0.2, f"Average Annual Growth Rate: {data_subset['Growth'].mean():.2f}%",
         horizontalalignment='center', verticalalignment='center', fontsize=16)
plt.axis('off')
plt.title('Summary Statistics for Type 2 Diabetes Cases (Age < 40)', fontsize=18)
plt.tight_layout()
```

ENSEMBLE- RandomForest,GradientBoost,XGBOOST

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor, VotingRegressor
from sklearn.metrics import mean_absolute_error, r2_score
from xgboost import XGBRegressor

# Load and prepare the data
data = pd.read_csv('NDA Young People with Type 2 Diabetes 2021-22 - Open Data v1.0.csv')
data_subset = data.iloc[57:117]

features = ['Section', 'Table/Figure', 'Title', 'Period', 'Country', 'Age_group', 'Breakdown/Measure', 'Data_type']
target = 'Value'

X = data_subset[features]
y = data_subset[target]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Preprocess data
preprocessor = ColumnTransformer(
    transformers=[
        ('cat', OneHotEncoder(handle_unknown='ignore'), features)
    ])

# Define the base models
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
gb_model = GradientBoostingRegressor(n_estimators=100, random_state=42)
xgb_model = XGBRegressor(n_estimators=100, random_state=42)

# Create the ensemble model
ensemble_model = VotingRegressor(
    estimators=[
        ('rf', rf_model),
        ('gb', gb_model),
        ('xgb', xgb_model)
    ]
)

# Create the full pipeline
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('imputer', SimpleImputer(strategy='mean')),
    ('regressor', ensemble_model)
])

# Fit the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate and print the performance metrics
mae = mean_absolute_error(y_test, y_pred)
xg_boost_r2 = r2_score(y_test, y_pred)

print(f"Mean Absolute Error: {mae:.4f}")
print(f"R-squared Score: {xg_boost_r2:.4f}")
```

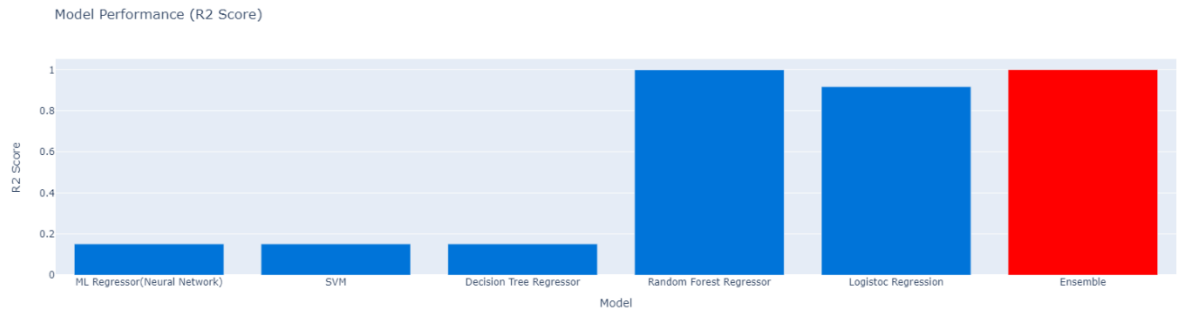
```
# Visualize the actual vs predicted values
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2)
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs Predicted Values (Ensemble Model)')
plt.tight_layout()
plt.show()

# Feature importance (using Random Forest as an example)
rf_model = model.named_steps['regressor'].estimators_[0]
feature_importance = rf_model.feature_importances_
feature_names = model.named_steps['preprocessor'].get_feature_names_out()

importance_df = pd.DataFrame({'feature': feature_names, 'importance': feature_importance})
importance_df = importance_df.sort_values('importance', ascending=False).head(10)

plt.figure(figsize=(12, 8))
sns.barplot(x='importance', y='feature', data=importance_df, palette='viridis')
plt.title('Top 10 Feature Importances (Random Forest in Ensemble)')
plt.xlabel('Importance')
plt.tight_layout()
plt.show()
```

ALL MODELS SCORE COMPARISION

```
[ ] !pip install dash
```

```
Requirement already satisfied: dash in /usr/local/lib/python3.10/dist-packages (2.17.1)
Requirement already satisfied: Flask<3.1,>=1.0.4 in /usr/local/lib/python3.10/dist-packages (from dash) (2.2.5)
Requirement already satisfied: Werkzeug<3.1 in /usr/local/lib/python3.10/dist-packages (from dash) (3.0.3)
Requirement already satisfied: plotly>=5.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (5.15.0)
Requirement already satisfied: dash-html-components==2.0.0 in /usr/local/lib/python3.10/dist-packages (from dash) (2.0.0)
```

```
import dash
from dash import dcc, html
from dash.dependencies import Input, Output
import plotly.express as px
import pandas as pd
import numpy as np

# Initial data
data = {
    'Model': ['ML Regressor(Neural Network)', 'SVM', 'Decision Tree Regressor', 'Random Forest Regressor', 'Ensemble','Logistoc Regression'],
    'R2 Score': [ML_regressor_r2, SVM_r2, DTR_r2, RFR_r2, xg_boost_r2,logistoc_regression_accuracy]
}

df = pd.DataFrame(data)

# Function to determine the color for each bar
def get_color(score, max_score):
    return 'rgb(255,0,0)' if score == max_score else 'rgb(0,116,217)'

app = dash.Dash(__name__)

app.layout = html.Div([
    html.H1("Model R2 Score Dashboard"),
    dcc.Graph(id='accuracy-graph'),
    html.Div(id='best-model-text')
])

@app.callback(
    [Output('accuracy-graph', 'figure'),
     Output('best-model-text', 'children')],
    Input('accuracy-graph', 'id')
)
def update_graph(_):
    max_score = df['R2 Score'].max()
    df['Color'] = df['R2 Score'].apply(lambda x: get_color(x, max_score))
    best_model = df.loc[df['R2 Score'] == max_score, 'Model'].values[0]

    fig = px.bar(df, x='Model', y='R2 Score', title='Model Performance (R2 Score)', color='Color', color_discrete_map="identity")
    fig.update_layout(yaxis_title='R2 Score', showlegend=False)

    best_model_text = f"Best performing model: {best_model} (R2 Score: {max_score:.4f})"

    return fig, best_model_text

if __name__ == '__main__':
    app.run_server(debug=True)
```



Best performing model: Ensemble (R2 Score: 0.9992)

CARDIFF METROPOLITAN UNIVERSITY
APPLICATION FOR ETHICS APPROVAL

When undertaking a research or enterprise project, Cardiff Met staff and students are obliged to complete this form in order that the ethics implications of that project may be considered.

If the project requires ethics approval from an external agency (e,g., NHS), you will not need to seek additional ethics approval from Cardiff Met. You should however complete Part One of this form and attach a copy of your ethics letter(s) of approval in order that your School has a record of the project.

The document ***Ethics application guidance notes*** will help you complete this form. It is available from the Cardiff Met website. The School or Unit in which you are based may also have produced some guidance documents, please consult your supervisor or School Ethics Coordinator.

Once you have completed the form, sign the declaration and forward to the appropriate person(s) in your School or Unit.

PLEASE NOTE:
Participant recruitment or data collection MUST NOT commence until ethics approval has been obtained.

PART ONE

Name of applicant:	Manisha
Supervisor (if student project):	Sarah May Mcvey
School / Unit:	CST
Student number (if applicable):	St20273663
Programme enrolled on (if applicable):	MSc in Data Science (Sep 2023)
Project Title:	Evaluating Prediction Methods for Diabetes Onset: A Comparative Study of Accuracy
Expected start date of data collection:	No requirement for new data collection, instead using existing benchmark dataset which is publicly available.
Approximate duration of data collection:	N/A
Funding Body (if applicable):	N/A
Other researcher(s) working on the project:	N/A
Will the study involve NHS patients or staff?	No
Will the study involve taking samples of human origin from participants?	No

Does your project fall entirely within one of the following categories:	
Paper based, involving only documents in the public domain	Yes

Laboratory based, not involving human participants or human tissue samples	No
Practice based not involving human participants (eg curatorial, practice audit)	No
Compulsory projects in professional practice (eg Initial Teacher Education)	No
A project for which external approval has been obtained (e.g., NHS)	No
If you have answered YES to any of these questions, expand on your answer in the non-technical summary. No further information regarding your project is required. If you have answered NO to all of these questions, you must complete Part 2 of this form	

In no more than 150 words, give a non-technical summary of the project
This project aims to predict the onset of Type 2 diabetes in young people using machine learning techniques based on clinical, demographic, and lifestyle data. Machine learning methods are increasingly utilized for diabetes prediction due to their high accuracy, automation capabilities, and ability to facilitate early intervention. Early detection of diabetes is crucial for preventing complications and improving patient outcomes. Machine learning algorithms can identify subtle risk factors that may not be easily noticeable to human observers, allowing for earlier and more effective intervention. The project involves developing predictive models using data from the National Diabetes Audit, specifically focusing on young people with Type 2 diabetes. This dataset is publicly available on the NHS Digital platform and contains comprehensive information on various factors influencing diabetes onset. By leveraging this dataset, the project aims to create and validate machine learning models that enhance the accuracy and efficiency of diabetes prediction, ultimately contributing to better patient care and management. Link: https://digital.nhs.uk/data-and-information/publications/statistical/national-diabetes-audit-yt2/young-people-with-type-2-diabetes-detailed-report-2021-22

DECLARATION: **I confirm that this project conforms with the Cardiff Met Research Governance Framework** **I confirm that I will abide by the Cardiff Met requirements regarding confidentiality and anonymity when conducting this project.** **STUDENTS: I confirm that I will not disseminate any material produced as a result of this project without the prior approval of my supervisor.**	
Signature of the applicant: 	Date: 21.07.2024

CARDIFF METROPOLITAN UNIVERSITY
APPLICATION FOR ETHICS APPROVAL

FOR STUDENT PROJECTS ONLY	
Name of supervisor: Sarah-May Mcvey	Date: 31/07/24
Signature of supervisor: Smcvey	

Research Ethics Committee use only	
Decision reached:	Project approved ☐ Project approved in principle ☐ Decision deferred ☐ Project not approved ☐ Project rejected ☐
Project reference number: Click here to enter text.	
Name: Click here to enter text.	Date: Click here to enter a date.
Signature:	
Details of any conditions upon which approval is dependant: Click here to enter text.	

PART TWO

A RESEARCH DESIGN	
A1 Will you be using an approved protocol in your project?	No
A2 If yes, please state the name and code of the approved protocol to be used[1]	
Click here to enter text.	
A3 Describe the research design to be used in your project	
This dissertation aims to predict Type 2 diabetes in young people using machine learning techniques based on clinical, demographic, and lifestyle data. The collected dataset will be split into training, validation, and testing subsets. The machine learning models will be built using various techniques such as logistic regression, decision trees, support vector machines, neural networks, and ensemble methods using the training dataset. These trained models will be validated using the validation dataset and then tested using the testing pool. Based on accuracy and other performance metrics, the best model will be selected to build an ensemble model, which combines the best-performing models to achieve higher accuracy in predicting Type 2 diabetes. Due to time constraints, this dissertation uses a benchmark dataset that is publicly available. The dataset, provided by the National Diabetes Audit and available on the NHS Digital platform, contains comprehensive clinical, demographic, and lifestyle data for young people with Type 2 diabetes. This	

[1] An Approved Protocol is one which has been approved by Cardiff Met to be used under supervision of designated members of staff; a list of approved protocols can be found on the Cardiff Met website here

dataset includes detailed and anonymized records, ensuring patient privacy while facilitating the development of predictive algorithms. The dataset will be preprocessed to remove personal information and will be used for training, validation, and testing purposes to develop robust and accurate machine learning models for early diabetes detection.

A4 Will the project involve deceptive or covert research?	No
A5 If yes, give a rationale for the use of deceptive or covert research	
Click here to enter text.	
A6 Will the project have security sensitive implications?	No
A7 If yes, please explain what they are and the measures that are proposed to address them	
This may affect some students undertaking research on Data Security. Complete this section if needed.	

B PREVIOUS EXPERIENCE
B1 What previous experience of research involving human participants relevant to this project do you have?
Yes, I have experience in building a predictive model for Type 2 diabetes onset, which involved analyzing clinical, demographic, and lifestyle data from patients. For this project, I used the publicly available National Diabetes Audit (NDA) dataset, focusing on young people with Type 2 diabetes. This dataset includes detailed medical records but did not involve direct human participation. My work concentrated on applying various machine learning techniques, including logistic regression, decision trees, support vector machines, neural networks, and ensemble methods, to predict diabetes risk factors and outcomes. Throughout the project, I ensured the privacy and integrity of the data while adhering to ethical research standards, including data anonymization and secure handling of sensitive information. This experience has equipped me with the necessary skills to responsibly manage medical data and apply advanced analytical techniques to improve disease prediction and patient care.
B2 **Student project only** What previous experience of research involving human participants relevant to this project does your supervisor have?
My supervisor, Dr. Sarah May Mcvey, has extensive 3 years of experience in research involving human participants.

C POTENTIAL RISKS
C1 What potential risks do you foresee?
There are no significant identified risks associated with this project. The National Diabetes Audit dataset is a benchmark, reputable, and publicly available dataset. Patients' personal information has been removed to ensure privacy and ethical compliance. This dataset has been used in numerous reputable publications, indicating its reliability and suitability for research purposes. Additionally, all data handling and analysis will adhere to strict ethical standards and guidelines, ensuring the protection of patient privacy and data integrity throughout the project.
C2 How will you deal with the potential risks?
N/A

When submitting your application you **MUST** attach a copy of the following:

- All information sheets
- Consent/assent form(s)

An exemplar information sheet and participant consent form are available from the Research section of the Cardiff Met website.

FOR INTERVIEWS AND FOCUS GROUP TYPE DATA COLLECTION

Cardiff Metropolitan University
Ethics Committee

PARTICIPANT CONSENT FORM

Cardiff Metropolitan University Ethics Reference Number:
Participant name or Study ID Number: St20273663
Title of Project: Evaluating Prediction Methods for Diabetes Onset: A Comparative Study of Accuracy
Name of Researcher: Manisha

Participant to complete this section: **Please initial each box.**

1. I confirm that I have read and understand the information sheet for the above study. I have had the opportunity to consider the information, ask questions and have had these answered satisfactorily. ☑

2. I understand that my participation is voluntary and that I am free to withdraw at any time, without giving any reason. ☑

3. I agree to take part in the above study. ☑

The following statements could also be included on the consent form if appropriate:

1. I agree to the interview / focus group / consultation being audio recorded ☑

2. I agree to the interview / focus group / consultation being video recorded ☑

3. I agree to the use of anonymised quotes in publications
   I agree to my quotes being attributed to me ☑

21-07-2024

_____ _____ _____
Signature of Participant Date

_____ _____
Name of person taking consent Date

_____
Signature of person taking consent

** When completed, 1 copy for participant & 1 copy for researcher site file*